Human-Understandable Explanations of Neural Networks

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

(Dr.-Ing.)

von der KIT-Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)
genehmigte

Dissertation

von

Anna Nguyen

Tag der mündlichen Prüfung:	23. November 2021
Referent:	Prof. Dr. York Sure-Vetter
Korreferent:	Prof. Dr. Heiko Paulheim

Danksagung

Zu Beginn meines Studiums habe ich mich mit der theoretischen Mathematik befasst. Schnell habe ich gemerkt, dass mich auch die Praxis sehr interessiert, weswegen ich mich im Verlauf meines Studiums der Wirtschaftsmathematik zuwandte. Dort entwickelte ich die Leidenschaft Daten zu analysieren. Durch die Möglichkeit einer Promotion konnte ich mein Wissen in der Informatik vertiefen, um Werkzeuge zu erlernen große Datenmengen zu analysieren. Während meiner dreijährigen Promotion an dem Lehrstuhl Web Science am AIFB konnte ich nicht nur mein Fachwissen in ein neues Feld erweitern, sondern bekam auch die Möglichkeit an einem der einschlägigsten Themen – der künstlichen Intelligenz – zu forschen. Dabei hat mich die Thematik der Erklärbarkeit immer fasziniert. Gerade mich als Quereinsteiger begeistert es, anderen Menschen neuronale Netze erklären zu können.

Ich danke meinem Doktorvater York Sure-Vetter dafür, dass er mir die Forschung in seiner Gruppe ermöglichte und für die Begleitung meiner Dissertation. Durch die Arbeit am Lehrstuhl habe ich in den drei Jahren viel gelernt und mich weiterentwickelt, sowohl inhaltlich als auch persönlich. Des Weiteren möchte ich meinem Korreferenten Heiko Paulheim für das Interesse an meinem Thema und sein wertvolles Feedback danken. Ferner danke ich Rudi Studer für sein wertvolles Feedback zu meinem Probevortrag und Beate Kühner für das Koordinieren der Termine für meine Verteidigung.

Außerdem bedanke ich mich bei allen Kollegen des Yordiverses, die mich bei meiner Dissertation unterstützt haben. Die Arbeitsatmosphäre war immer freundschaftlich und familiär. Der inhaltiche Austausch war stets konstruktiv und hat meine Forschung vorangetrieben. Besonderer Dank gilt meinem langjährigen Bürokollegen

Tobias Weller für die gemeinsame Arbeit an verschiedenen Forschungspapieren und die stetige Unterstützung während meiner Promotion.

Einen besonderen Dank an meine Studenten und Koautoren Adrian Oberföll, Franz Krause und Daniel Hagenmayer mit deren mitwirken ich meine Forschung voranbringen konnte. Ich danke meinen wissenschaftlichen Hilfskräften Jannik Deuschel und Yanglin Xu für ihre Arbeit, die meine Forschung unterstützten.

Des Weiteren danke ich Guido Brückner für die Unterstützung während meiner Promotion. Sein Feedback war immer kritisch und anspruchsvoll, aber stets konstruktiv und hilfreich. Ich danke Fenja Scheu für die Motivation beim Aufschrieb der Dissertation durch die gemeinsamen Tage im Büro. Außerdem danke ich Cindy Hong und Chau Phuc Nguyen für die Korrektur meiner Arbeit.

Abschließend danke ich meiner Familie und meinen Freunden, die stets an mich glaubten und mir Durchhaltevermögen gegeben haben.

Deutsche Zusammenfassung

Das 21. Jahrhundert ist durch Datenströme enormen Ausmaßes gekennzeichnet. Dies hat die Popularität von Berechnungsmodellen, die sehr datenintensiv sind, wie z.B. neuronale Netze, drastisch erhöht. Aufgrund ihres großen Erfolges bei der Mustererkennung sind sie zu einem leistungsstarken Werkzeug für Vorhersagen, Klassifizierung und Empfehlungen in der Informatik, Statistik, Wirtschaft und vielen anderen Disziplinen geworden. Trotz dieser verbreiteten Anwendung sind neuronale Netze Blackbox-Modelle, d.h. sie geben keine leicht interpretierbaren Einblicke in die Struktur der approximierten Funktion oder in die Art und Weise, wie die Eingabe in die entsprechende Ausgabe umgewandelt wird. Die jüngste Forschung versucht, diese Blackboxen zu öffnen und ihr Innenleben zu enthüllen. Bisher haben sich die meisten Forschungsarbeiten darauf konzentriert, die Entscheidungen eines neuronalen Netzes auf einer sehr technischen Ebene und für ein Informatikfachpublikum zu erklären. Da neuronale Netze immer häufiger eingesetzt werden, auch von Menschen ohne tiefere Informatikkenntnisse, ist es von entscheidender Bedeutung, Ansätze zu entwickeln, die es ermöglichen, neuronale Netze auch für Nicht-Experten verständlich zu erklären. Das Ziel ist, dass Menschen verstehen können, warum das neuronale Netz bestimmte Entscheidungen getroffen hat, und dass sie das Ergebnis des Modells durchgehend interpretieren können.

Diese Arbeit beschreibt ein Rahmenwerk, das es ermöglicht, *menschlich verständliche Erklärungen für neuronale Netze* zu liefern. Wir charakterisieren menschlich nachvollziehbare Erklärungen durch sieben Eigenschaften, nämlich Transparenz, Überprüfbarkeit, Vertrauen, Effektivität, Überzeugungskraft, Effizienz und Zufriedenheit. In dieser Arbeit stellen wir Erklärungsansätze vor, die diese Eigenschaften erfüllen. Zunächst stellen wir *TransPer* vor, ein Erklärungsrahmenwerk für neuronale Netze, insbesondere für solche, die in Produktempfehlungssystemen verwendet werden. Wir definieren Erklärungsmaße auf der Grundlage der Relevanz der Eingaben, um die Vorhersagequalität des neuronalen Netzes zu analysieren und KI-Anwendern bei der Verbesserung ihrer neuronalen Netze zu helfen. Dadurch werden Transparenz und Vertrauen geschaffen. In einem Anwendungsfall für ein Empfehlungssystem werden auch die Überzeugungskraft, die den Benutzer zum Kauf eines Produkts veranlasst, und die Zufriedenheit, die das Benutzererlebnis angenehmer macht, berücksichtigt. Zweitens, um die Blackbox des neuronalen Netzes zu öffnen, definieren wir eine neue Metrik für die Erklärungsqualität *ObAlEx* in der Bildklassifikation. Mit Hilfe von Objekterkennungsansätzen, Erklärungsansätzen und ObAlEx quantifizieren wir den Fokus von faltenden neuronalen Netzwerken auf die tatsächliche Evidenz. Dies bietet den Nutzern eine effektive Erklärung und Vertrauen, dass das Modell seine Klassifizierungsentscheidung tatsächlich auf der Grundlage des richtigen Teils des Eingabebildes getroffen hat. Darüber hinaus ermöglicht es die Überprüfbarkeit, d. h. die Möglichkeit für den Benutzer, dem Erklärungssystem mitzuteilen, dass sich das Modell auf die falschen Teile des Eingabebildes konzentriert hat. Drittens schlagen wir *FilTag* vor, einen Ansatz zur Erklärung von faltenden neuronalen Netzwerken durch die Kennzeichnung der Filter mit Schlüsselwörtern, die Bildklassen identifizieren. In ihrer Gesamtheit erklären diese Kennzeichnungen die Zweckbestimmung des Filters. Einzelne Bildklassifizierungen können dann intuitiv anhand der Kennzeichnungen der Filter, die das Eingabebild aktiviert, erklärt werden. Diese Erklärungen erhöhen die Überprüfbarkeit und das Vertrauen. Schließlich stellen wir *FAIRnets* vor, das darauf abzielt, Metadaten von neuronalen Netzen wie Architekturinformationen und Verwendungszweck bereitzustellen. Indem erklärt wird, wie das neuronale Netz aufgebaut ist werden neuronale Netzer transparenter; dadurch dass ein Nutzer schnell entscheiden kann, ob das neuronale Netz für den gewünschten Anwendungsfall relevant ist werden neuronale Netze effizienter.

Alle vier Ansätze befassen sich mit der Frage, wie man Erklärungen von neuronalen Netzen für Nicht-Experten bereitstellen kann. Zusammen stellen sie einen wichtigen Schritt in Richtung einer für den Menschen verständlichen KI dar.

Abstract

The 21st century is characterized by an influx of tremendous amounts of data. This has dramatically increased the popularity of computational models that are very data-intensive such as neural networks. Due to the great success in pattern recognition, they have become a powerful tool for example in prediction, classification, and recommendation in computer science, statistics, economics, and many other disciplines. Despite this widespread use, neural networks are black-box models, meaning that they do not give any readily interpretable insights into the structure of the approximated function or into how input is transformed into its corresponding output. Recent research has attempted to pry open these black boxes and reveal their inner workings. So far, most research has focused on explaining decisions of a neural network at a highly technical level and to a computer science expert audience. As neural networks become more widely deployed, including by people without a computer science background, it is crucial to develop approaches that allow for explanations of neural networks understandable to non-experts. The goal is that humans can understand why certain decisions were made by the neural network and can consistently interpret the model's result.

This work describes a framework to provide *human-understandable explanations of neural networks*. We characterize human-understandable explanations by seven properties, namely transparency, scrutability, trust, effectiveness, persuasiveness, efficiency, and satisfaction. In this work, we present explanation approaches that satisfy these properties. First, we present *TransPer*, an explanatory framework for neural networks, in particular those used in product recommender systems. We define explanation measures based on input relevancies to understand the neural network's prediction quality and to help AI practitioners improve their neural network. This captures transparency and trust. Additionally, in a recommendation system use case, persuasion, which persuades the user to buy a product, and satisfaction, which makes the user experience more pleasant, are also included. Second, to open the neural network black box, we define a new explanation quality metric *ObAlEx* for image classification. Using object detection approaches, explanation approaches, and ObAlEx, we quantify the focus of Convolutional Neural Networks on the actual evidence. This provides the users an effective explanation and trust, i.e., that the model has indeed made its classification decision based on the correct part of the input image. Furthermore, it enables scrutability, i.e., the capability for the user to declare that the model has focused on the wrong parts of the input image. Third, we propose *FilTag*, an approach to explain Convolutional Neural Networks by tagging the filters with keywords that identify classes of images. In aggregate, these tags explain what the filter does. Individual image classifications can then be intuitively explained in terms of the tags of the filters that the input image activates. These explanations enhance scrutability and trust. Finally, we present *FAIRnets*, which aims to process metadata such as architecture information and intended use. This makes neural networks more transparent, i.e., to explain the neural network's architecture, which problems they solve, and so on, and more efficient, i.e., to help the user quickly decide whether the neural network is relevant for their intended use case.

All four approaches address the question of how to generate explanations of neural networks for non-experts. Together, they constitute an important step in the direction of human-understandable AI.

Contents

1 Introduction

Artificial Intelligence (AI) has gained importance in recent years due to the availability of data and computing power. However, these approaches are so-called black-box models that do not explain their results. Therefore, there has been an increase in methods in the area of explanatory AI. However, the approaches developed so far are very technical and intended for computer scientists. As more and more non-computer scientists use neural networks, it is even more crucial to develop explanations for this audience. In this work, we will tackle questions like "How exactly does a neural network recognize an object in an image?" and "Why does a recommender system recommend exactly this product?".

1.1 Motivation

The need for explainable AI can be demonstrated by three simple examples. Neural networks have become very popular in the field of computer vision. They yield very high accuracy in different tasks such as image classification and object recognition. Despite this achievement, the state of the art research has revealed that some neural networks classify correctly but due to wrong features of the image. Ribeiro, Singh, and Guestrin, for example, did an experiment to show which features were relevant for the decision of an image classifier. They trained a classifier that distinguished pictures of wolves and huskies. They intentionally used images of wolves with snow in the background to train the network [RSG16]. When inputting an image of a husky with snow in the background, the neural network classifies it as a wolf. When

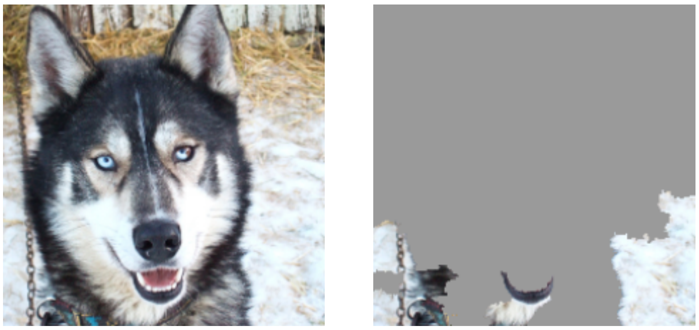


Figure 1.1: Example of a classifier trained on images of wolves on snowy background taken from [RSG16]. On the left-hand side is an image of a husky which is classified as wolf. On the right-hand side is a visualization of the most important pixels as an explanation which is the snowy background.

visualizing the important features, they could see that the neural network focused on the background to make the decision, see Fig. 1.1. Another example of classifying right for the wrong reasons can be seen in Fig. 1.2. Here, the neural network classifies the object in the images correctly due to the focus on the source tags (see Fig. 1.2 (a) and (c)) [Lap+19]. When removing the source tag in Fig. 1.2 (b) and (d), the neural network classifies wrong and has no specific focus. These experiments show how important it is that the neural network focuses on the right features and that it is crucial to gain explanations for the neural network's decision.

Another popular application of neural networks is in e-commerce. There, they are used to make personalized product recommendations. For example, Amazon recommends new products to buy, Netflix recommends movies to watch, and Facebook runs personalized ads. Neural networks are a preferred tool because they can capture the specific interests of the users and, thus, personalize the recommendations. Therefore, they substantially improve the relevance of the recommendations. The European Union has found it necessary to regulate access to personal data to protect individuals (GDPR) [PE16]. In particular, they require that every user can see how their data is being used. This means, for example, that every customer at Amazon can see their data usage. In this context, it is especially important to know how the (training) data is used in the recommender system. It is all the more crucial that the explanations are understandable, especially for laypersons, since they are directly affected by neural networks.

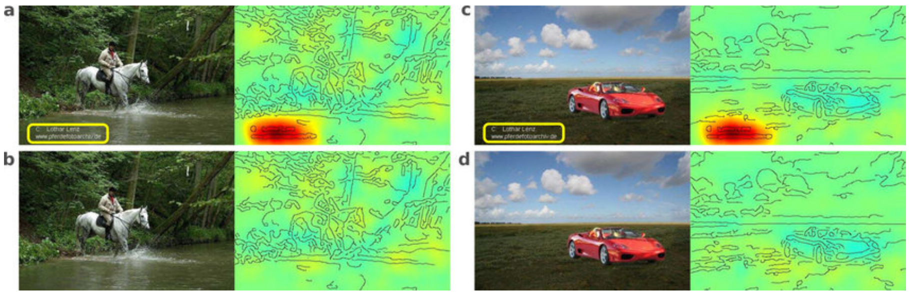


Figure 1.2: Example of classifying right for the wrong reasons taken from [Lap+19]. On the left-hand side is the image to classify and on the right-hand side a visualization of the most important pixels as explanation. (a) and (c) are correctly classified whereas (b) and (d) are wrongly classified.

1.2 Problem Statement

The main research question that we seek to answer in this thesis is:

Research Question. *How to make explanations human-understandable?*

However, what are explanations that are understandable to humans? Imagine the following situation. You want to explain to a child what a house is. You start by describing its parts such as door, windows and roof but the child is not satisfied and wants to know why these components describe a house. Then you continue to describe its purposes such as the door being the entrance to the house. The next question follows up: why? Because you live in a house and you want to enter it. After the third why you don't know anymore. Some researchers have been confronted with this situation. Biran and Cotton have come to the following conclusion [BC17]:

> "Explanation is closely related to the concept of interpretability: systems are interpretable if their operations can be understood by a human, either through introspection or through a produced explanation. In the case of machine learning models, explanation is often a difficult task since most models are not readily interpretable."

To address this problem in this thesis, we first define *explainability* as:

Definition (Explainability)**.** *Explainability is the degree to which a human*

1. *can understand the cause of a decision.*
2. *can consistently interpret the model's result.*

Now, we have a definition but how can we measure explainability? Tintarev and Masthoff [TM12] develop seven objectives that understandable explanations should address for people. An explanation approach should be an extension to a model that enables the following features:

- *Transparency* explains how the system works. For example, convolutional neural networks in image classification have filters that highlight certain patterns. Just knowing this explains to a certain extent how the neural network works.
- *Scrutability* allows users to tell the system it is wrong. This relates to our motivation with the example of the wolf and the husky. There, we could tell that the snowy background is wrong.
- *Trust* increases the users' confidence in the system. Consider the example with the source tags. If the user knows that the neural network classifies correctly based on the actual object and not the source tag that would increase the user's confidence.
- *Effectiveness* helps users to make good decisions. Consider for example our above motivation with classifying an image right for the right reasons.
- *Persuasiveness* convinces users to try. For example, in an online shop, an explanation of the neural network suggesting products would additionally convince people to look at that product.
- *Efficiency* helps users to make decisions faster. For example, we have a problem, but we do not know which neural network is suitable. With an explanation of what which network does, we would come to a faster choice of a suitable neural network.
- *Satisfaction* increases the ease of usability or enjoyment. Consider the online shop example. An explanation of the product recommendations helps the user to understand the suggestions and to act on them. This leads to more user-friendliness.

These features make the potentially vague term of *explainability* more rigorously tractable. In this thesis, we will introduce a framework that addresses these seven goals to make explanations human-understandable.

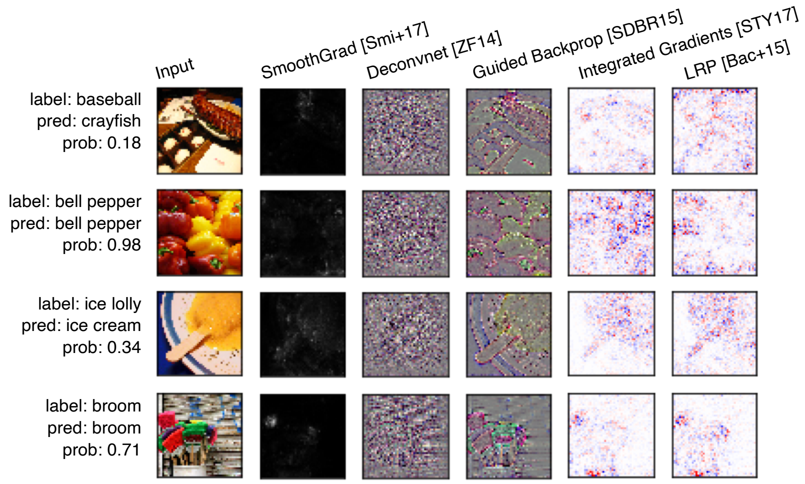


Figure 1.3: Exemplary explanation approaches taken from [Alb+19].

1.3 State of the Art

In *eXplainable Artificial Intelligence* (XAI), a lot has happened in the last few years due to the boom of AI, especially with neural networks. Although there existed methods on how to build an interpretable neural network back in the 90s [GHMS92, LC99, AW93], the trend has resurfaced from 2010 onwards. Most explanation approaches of neural networks occur in the field of computer vision. There, explanation approaches visualize important features to bring the explanation closer to the human being. For example in image classification, perturbation-based approaches perturb image features such as image areas or image colors to find out which features are relevant. For example, *Deconvnet* gradually covers areas of an image to find out which area was most relevant [ZF14] (see Fig. 1.3). Another method is *LIME* [RSG16] which extends Deconvnet. It divides the input image into similar color areas, called superpixels. Then, the weighting of the superpixels is calculated to determine the most important areas in the image for the classification. With these approaches, however, the size of the window that one covers is relevant. This can lead to artificially optimized explanation approaches, i.e., the window size is selected in a way that the object to be classified lies within it. Gradient-based approaches on the other hand examine the derivative to explain a neural network. For example, the *Gradient-based Sensitivity Analysis* can be used to evaluate how sensitive the predictions react to small changes in input features [RHW86]. Input values that lead to an increase or decrease in the predictions can be decisive for a certain class [SVZ14]. Known applications include *SmoothGrad*, which adds noise to a picture to find the relevant pixels [Smi+17], and *Integrated Gradients*, which combines sensitivity analysis and implementation invariance [STY17]. These methods show whether an input has a positive or negative influence on the prediction. However, it is not possible to make a quantitative statement because they do not preserve the former values. Other methods with a similar goal are backpropagation-based approaches which use attention mechanisms to see if the most important features are the desired ones. Known approaches are *Guided Backpropagation* [SDBR15] and *Layerwise Relevance Propagation* [Bac+15, Mon+19, MSM18]. Additional information on these methods can be found in Chapter 2. Fig. 1.3 shows some examples of those explanation approaches. Despite the number of visualization techniques, most of the time these explanations are aligned to the view of a computer scientist. Without a technical background, it is difficult to understand the meaning of the explanations. Thus, these approaches rely on a human expert who evaluates the explanations. With thousands of images, it can take some time to examine whether it corresponds to the desired explanation. Furthermore, most explanations do not provide any indication of how the training process or architecture should be changed to get an improvement.

1.4 Hypotheses

As we have seen in the previous section, the existing approaches go very deep in their explanations. This allows a more detailed explanation of the architecture of the neural network, for example, the individual neurons [Smi+17] or layers [Bac+15]. As illustrated by Figure 1.3, such explanations are targeted at computer science experts. We aim to bridge the gap between AI and (non-expert) humans. In particular, we want to develop approaches that make neural network explanations comprehensible to people without a computer science background. To this end, we formulate the following hypotheses.

Hypothesis 1. *Quantifying explanations based on the relevance of the input features facilitates the evaluation of not only the input data but also the neural network.*

Associated with Hypothesis 1 are the following research questions we want to tackle:

1. What parameters are relevant for understanding explanation quality?
2. Which implications can be derived from these relevancies?
3. Can these relevancies be used to improve the neural network (and if so, how)?

With the first hypothesis, we want to improve an existing explanatory approach by quantifying its explanations and, thus, making them more tangible for laypersons. In the second hypothesis, we turn to the neural network itself. How must a neural network be adapted so that we get the desired explanation (making it right for the right reasons)? Or what is the desired explanation, taking into account existing explanation approaches?

Hypothesis 2. *In image classification, object-aligned explanations can sufficiently map the desired explanation of humans and guarantee an intuitive explanation.*

Associated with Hypothesis 2 are the following research questions:

1. How can right for the right reasons be measured?
2. Does it satisfy the concept of classifying right for the right reasons?
3. Can it be included in the training process?

To break down the architecture of a neural network even further, we make use of semantics to give another dimension of explanation besides visualization. The third hypothesis considers filters in convolutional neural networks.

Hypothesis 3. *Filters in CNNs encode semantic information of the input images.*

Associated with Hypothesis 3 are the following research questions:

1. How can the encoded semantic information of the filters be decoded?
2. How precise are the tagged filters in predicting and understanding the output of the CNN (compared to visual methods)?
3. What benefit does link tagged filters to knowledge graphs offer?

Our first three hypotheses aim at complementing existing explanatory approaches and thus making them more comprehensible to humans. In doing so, we follow the state of the art by going deeper into the architecture of neural networks. However, the existing approaches still need to be evaluated by computer science experts. Our distinction from existing approaches is that we exploit existing approaches and enhance them with a measure to make them more tangible to non-experts. Furthermore, we make use of semantics to include another dimension of explanation. We note that all of these approaches, as well as those mentioned in Section 1.3 are what we consider *deep*, i.e., they take one neural network and one input and produce one explanation of the output.

In contrast, one may also consider explanations that are *broad*, i.e., ones that take into consideration multiple models at once. There is not only an unprecedented availability of data but also an unprecedented availability of statistical models such as neural networks that were trained on that data. A broad explanation approach can serve as a very first step in finding suitable models for the task at hand. What if you have an idea of a problem, but you do not know what kind of network to use? You would need a more general explanation about existing neural networks. The information basis on neural networks is quite large with repositories, containing neural networks, data sets for neural networks, and frameworks to build neural network architecture. It seems as if it is easy to find a neural network for every use case. However, when trying to find a suitable neural network, the search list is overwhelming positively and negatively. We can find a lot about neural networks, but the information still needs to be sorted or is redundant according to use cases. The availability is mostly not given or the repositories only provide a simple example. The detailed search is meager and not specific enough for explicit neural networks. The problem is the amount and creation speed of new information, the importance of which must be determined and compiled in a structured manner. Only then can people use and understand it. We address this problem in the final hypothesis by preparing neural networks according to the *FAIR Principles* [Wil+16, Wis+19, DPGK19] using semantics. The FAIR Principles give the guideline on how to make data findable, accessible, interoperable and reusable. To our knowledge, this type of broad explanation has not yet been addressed.

Hypothesis 4. *Neural network models which comply with the FAIR Principles support humans in dealing with the increase in volume, complexity, and creation speed of these models.*

Associated with Hypothesis 4 are the following research questions:

1. Which information should be provided about the neural network models to enable transparency?
2. Which information must be provided to apply an existing model to a novel use case?
3. What kind of functions should be provided for supporting humans in reusing neural networks?

Figure 1.4 visualizes the hypotheses and research questions of this thesis.

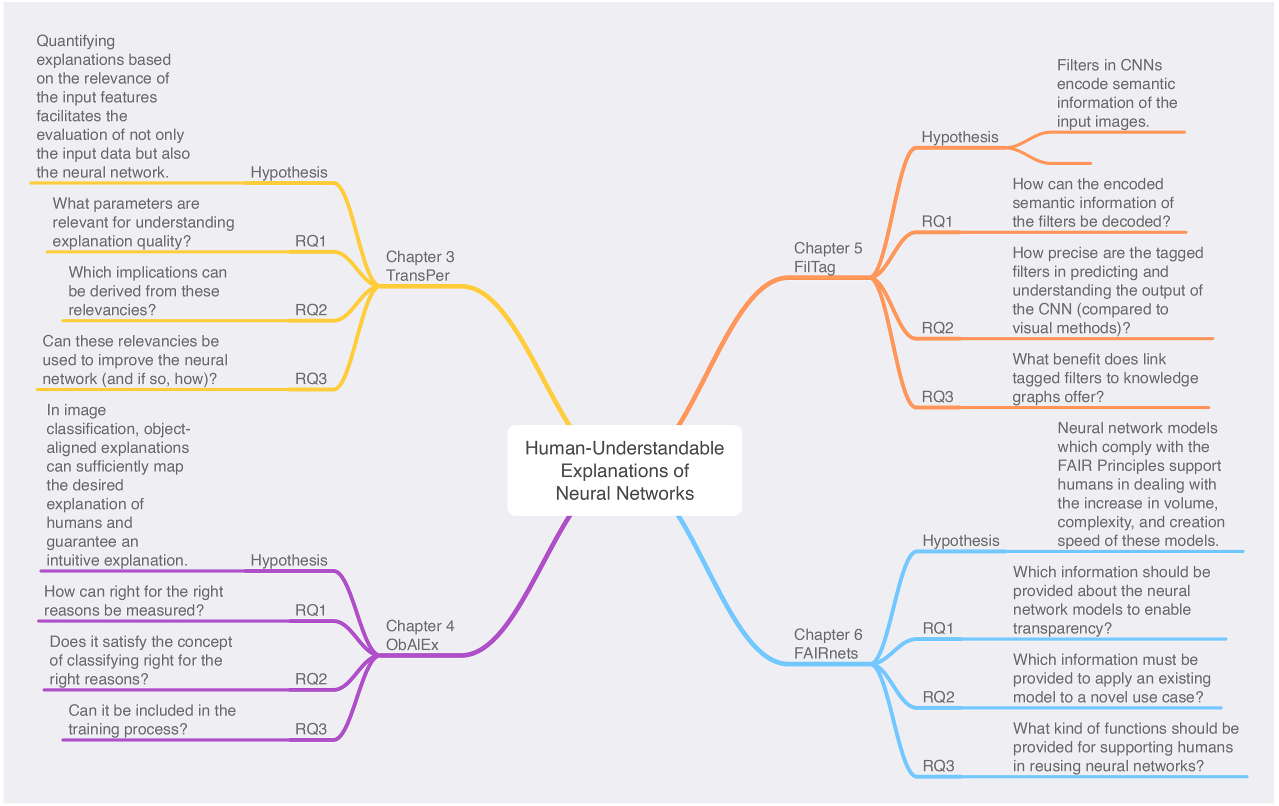

Human-Understandable Explanations of Neural Networks
Chapter 3 TransPer
Hypothesis
Quantifying explanations based on the relevance of the input features facilitates the evaluation of not only the input data but also the neural network.
RQ1
What parameters are relevant for understanding explanation quality?
RQ2
Which implications can be derived from these relevancies?
RQ3
Can these relevancies be used to improve the neural network (and if so, how)?
Chapter 4 ObAIEx
Hypothesis
In image classification, object-aligned explanations can sufficiently map the desired explanation of humans and guarantee an intuitive explanation.
RQ1
How can right for the right reasons be measured?
RQ2
Does it satisfy the concept of classifying right for the right reasons?
RQ3
Can it be included in the training process?
Chapter 5 FilTag
Hypothesis
Filters in CNNs encode semantic information of the input images.
RQ1
How can the encoded semantic information of the filters be decoded?
RQ2
How precise are the tagged filters in predicting and understanding the output of the CNN (compared to visual methods)?
RQ3
What benefit does link tagged filters to knowledge graphs offer?
Chapter 6 FAIRnets
Hypothesis
Neural network models which comply with the FAIR Principles support humans in dealing with the increase in volume, complexity, and creation speed of these models.
RQ1
Which information should be provided about the neural network models to enable transparency?
RQ2
Which information must be provided to apply an existing model to a novel use case?
RQ3
What kind of functions should be provided for supporting humans in reusing neural networks?

Figure 1.4: The hypotheses and research questions.

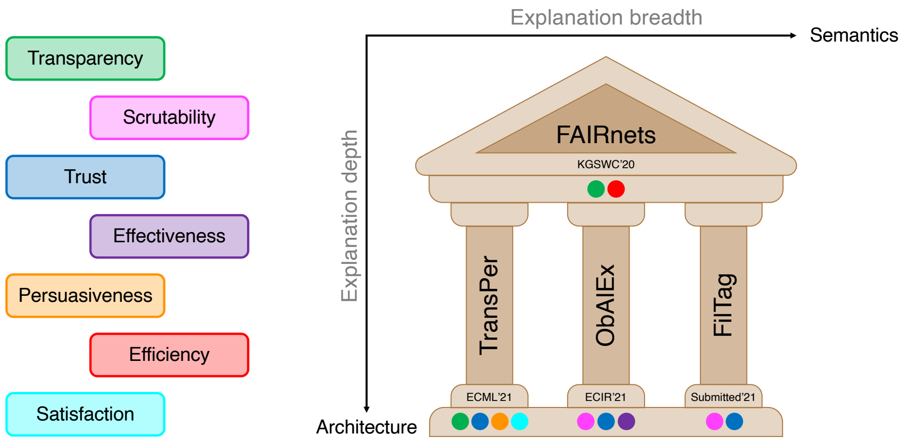


Figure 1.5: Overall framework.

1.5 Contributions

Considering the aforementioned hypotheses, our contribution is fourfold. *TransPer* deals with the question: "I just bought a dishwasher, why is another dishwasher recommended to me?" *ObAlEx* ensures that images of Huskies should not be recognized by snowy backgrounds. *FilTag* addresses the question: "How is this Picasso painting classified as a parrot?" *FAIRnets* tackles the hypothesis: "To apply machine learning at scale, data must be discoverable, accessible, interoperable, and reusable." Fig. 1.5 shows an overview of our contributions and categorizes them accordingly in a framework. If we consider the architecture axis, we see that TransPer, ObAlEx and FilTag examine the architecture of a neural network regarding the depth of explanation. That means, they drill down into the architecture. However, they are narrower in their breadth of explanation than FAIRnets because they explicitly refer to an existing neural network. Fairnets, on the other hand, does not go as deep as the other approaches in its explanations but is rather on the surface. Nevertheless, FAIRnets looks at the whole volume of existing neural networks, i.e. the explanations are broad. If we turn to the semantic axis, we see that FAIRnets and FilTag use semantic tools, while TransPer and ObAlEx do not use semantics. In the following, we will go into more detail on the individual contributions:

Contribution to Hypothesis 1. *TransPer, an explanatory framework for neural networks based on input relevancies and explanation quantity measures to evaluate the helpfulness of the resulting explanations.*

Firstly, we present *TransPer*, an explanatory framework for neural networks to provide transparency. It uses novel explanation measures based on *Layer-Wise Relevance Propagation* and can handle heterogeneous data and complex neural network architectures, such as combinations of multiple neural networks into one larger architecture. We apply and evaluate our framework on two real-world online shops. We show that the explanations provided by TransPer help understand recommendation quality, find new ideas on how to improve the recommender system, help the online shops understand their customers, and meet legal requirements such as the ones mandated by GDPR.

Contribution to Hypothesis 2. *ObAlEx, an object-aligned explanation quality metric for image classification.*

Secondly, to open the neural network black box, we define a new explanation quality metric *ObAlEx* for image classification. The effectiveness of Convolutional Neural Networks (CNNs) in classifying image data has been thoroughly demonstrated. To explain the classification to humans, methods for visualizing classification evidence have been developed in recent years. These explanations reveal that sometimes images are classified correctly, but for the wrong reasons, e.g. recall the example of wolves being recognized based on snowy backgrounds, or the example of the objects being recognized based on source tags from the beginning of this chapter. Of course, images should be classified correctly for the right reasons, i.e., based on the actual evidence. To this end, ObAlEx measures *ob*ject-*al*igned *ex*planation. Using object detection approaches, explanation approaches, and ObAlEx, we quantify the focus of CNNs on the actual evidence. Moreover, we show that additional training of the CNNs can improve the focus of CNNs without decreasing their accuracy and, thus, classify right for the right reasons.

Contribution to Hypothesis 3. *FilTag, an approach to explain the role of each convolutional filter of a convolutional neural network to non-technical-experts.*

Thirdly, we propose *FilTag*, an approach to effectively explain Convolutional Neural Networks even to non-experts by tagging the most activated filters with words describing objects from everyday life. The idea is that when images of a class frequently activate a convolutional filter, then that filter is tagged with that class. These tags explain what the filter does. Further, individual image classifications can then be intuitively explained in terms of the tags of the filters that the input image activates. Combining the tags with knowledge graphs gives a wider understanding of the filters. Finally, we show that the tags help analyze classification mistakes and that they can be further processed by computers.

Contribution to Hypothesis 4. *FAIRnets, an ontology for presenting neural networks following the FAIR principles and a knowledge graph representing over 18,400 publicly available neural networks.*

Finally, we present *FAIRnets*, an approach that processes the metadata to explain. Therefore, we define and build the neural network ontology *FAIRnets Ontology*, an ontology to make existing neural network models *findable*, *accessible*, *interoperable*, and *reusable* according to the FAIR principles. Our ontology allows us to model neural networks on a meta-level in a structured way, including the representation of all network layers and their characteristics. Based on that, we have modeled over 18,000 neural networks from GitHub, which we provide to the public as a knowledge graph called *FAIRnets*, ready to be used for recommending suitable neural networks to AI practitioners.

1.6 Outline

The thesis is structured as follows. After introducing fundamentals and concepts in Chapter 2, we go on to present our four approaches that shape our framework. We begin with TransPer in Chapter 3, an explanation approach for neural networks with a focus on product recommendation. Then, we propose two explanation approaches for computer vision to be more precise image classification, namely ObAlEx and FilTag in Chapter 4 and Chapter 5, respectively. Afterward, in Chapter 6, we propose FAIRnets, an approach to provide FAIR neural networks. Then, we conclude with a summary of our framework, including limitations and future work in Chapter 7. An outline of the thesis can be found in Figure 1.6.

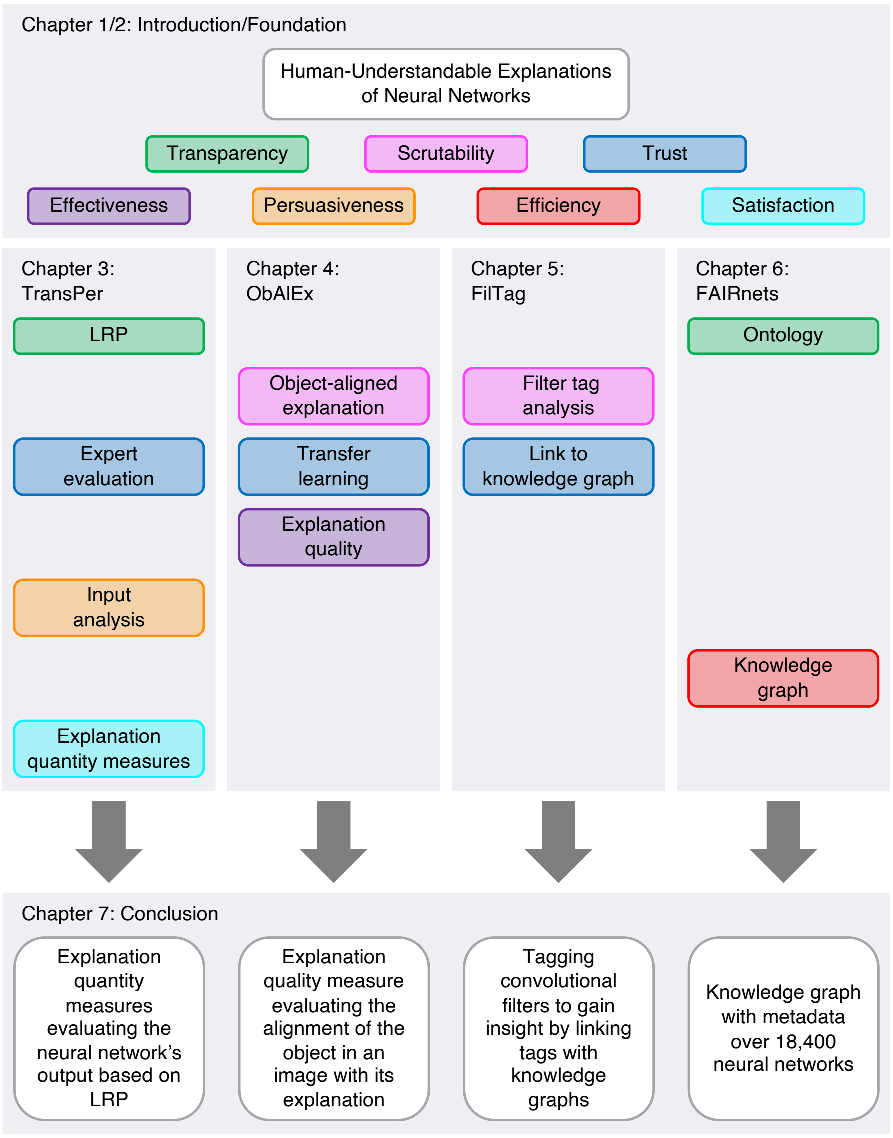

Chapter 1/2: Introduction/Foundation
Human-Understandable Explanations of Neural Networks
Transparency
Scrutability
Trust
Effectiveness
Persuasiveness
Efficiency
Satisfaction
Chapter 3: TransPer
LRP
Expert evaluation
Input analysis
Explanation quantity measures
Chapter 4: ObAIEx
Object-aligned explanation
Transfer learning
Explanation quality
Chapter 5: FilTag
Filter tag analysis
Link to knowledge graph
Chapter 6: FAIRnets
Ontology
Knowledge graph
Chapter 7: Conclusion
Explanation quantity measures evaluating the neural network's output based on LRP
Explanation quality measure evaluating the alignment of the object in an image with its explanation
Tagging convolutional filters to gain insight by linking tags with knowledge graphs
Knowledge graph with metadata over 18,400 neural networks

Figure 1.6: The outline of the thesis.

2 Foundation

In this chapter, we introduce notations, describe basic concepts and methods used throughout the thesis. After introducing some famous neural networks which we reuse in our approaches, we introduce basic explanation approaches.

2.1 Neural Network

Neural networks in computer science are a machine learning method in which a computer learns to perform a task by analyzing training examples. Modeled after the human brain, a neural network consists of thousands of *neurons* which are connected. It can be visualized as a direct graph, where the nodes are the neurons. The neurons each belong to one *layer* and are connected if so to the neurons in the next layer. These layers consist of an input, an output, and hidden layers in between, see Fig. 2.1. The edges here are *weights* that are determined during training. Each neuron has an *activation* which is calculated from an *activation function*, weight and activation of the previous neuron. In each training step, the weights are adjusted by *backpropagation*.

In this thesis, we consider trained neural networks, i.e. the weights are already determined. Following [NKHF21], consider a trained neural network with $K \in \mathbb{N}$ layers, i.e. hidden layers. Fig. 2.1 shows an example of a neural network architecture with $K = 2$. We refer to Π_k as the set of all neurons in the k-th layer, σ as a nonlinear monotonously increasing activation function, z_i^k as the activation of the i-th neuron in the k-th layer, $w_{ij}^{k,k+1}$ as the weight between the neurons z_i^k and z_j^{k+1}, and b_j^k as the *bias* term w.r.t. z_j^{k+1}. Assuming that we know the activations in Π_k, the activations

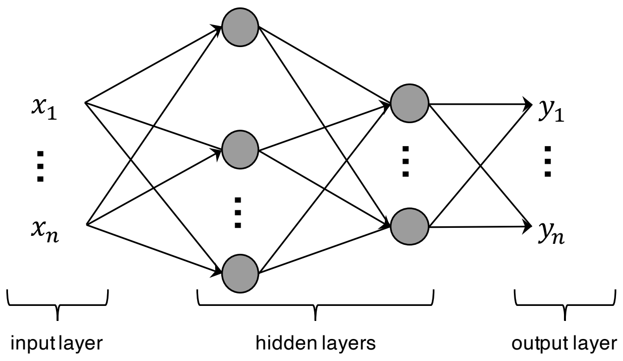


Figure 2.1: An exemplary neural network.

in Π_{k+1} can be determined via forward pass as follows:

$$z_j^{k+1} = \sigma\left(\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right) + b_j^k\right)$$

For non-connected neurons z_i^k and z_j^{k+1} we assume $w_{ij}^{k,k+1} = 0$. If a network has no bias, then $b_j^k = 0$. This is the basic architecture of a feed-forward neural network. In addition, there are also variations regarding the layers, which are presented hereafter.

Convolutional Neural Network (CNN)

Convolutional Neural Networks are mostly used in the field of image recognition because they are space invariant. A CNN has a *convolutional layer* as a characteristic feature which is an application of a *filter* on an input resulting in a *feature map*. The filters can detect specific patterns in an image that are relevant for certain prediction classes.

Fig. 2.2 shows such a convolution. First, the input image is converted into a digital image, more precisely an RGB image. By applying a filter that highlights specific features a feature map is created. This feature map summarizes detected features from the input. Usually, several filters are applied in parallel generating several feature maps that are used as input in the next convolutional layer. This parallelization is beneficial to learn complex patterns in an image. As the filters are learned during training, they are also the weights of the neural network. This is important when it comes to explanation approaches that visualize the weights to understand what the neural network does.

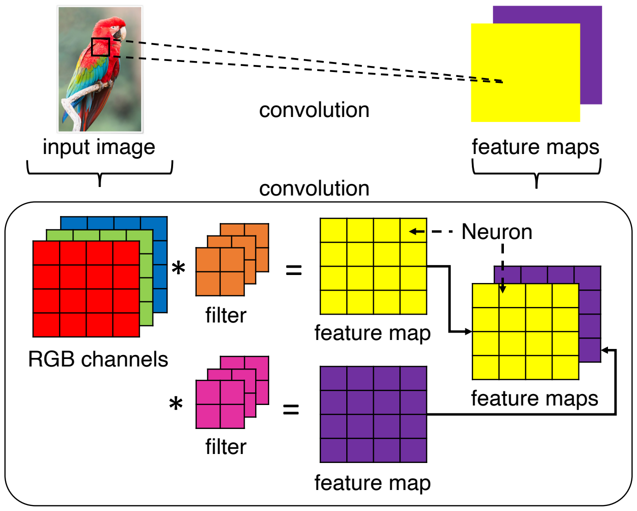


Figure 2.2: An exemplary convolution taken from [NHWF21].

Table 2.1: Overview of the CNNs used taken from [Ker]. The time is per inference step.

Model	Top-1 Accuracy	Top-5 Accuracy	Depth	Time (GPU)
VGG16	0.713	0.901	23	4.16ms
VGG19	0.713	0.900	26	4.38ms
ResNet50	0.749	0.921	-	4.55ms
InceptionV3	0.779	0.942	159	6.64ms
MobileNet	0.704	0.895	88	3.44ms

The neural network models we use throughout this thesis are pre-trained and taken from [Ker]. Table 2.1 shows an overview of the characteristics. In the following we will explain their architecture:

VGG. The architecture *VGG* was introduced by Simonyan and Zisserman. In their paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”, they examined the role of the depth of a CNN. They found that a depth of 16–19 layers gave the best accuracy in image recognition [SZ15]. Fig. 2.3 shows VGG16 with sixteen

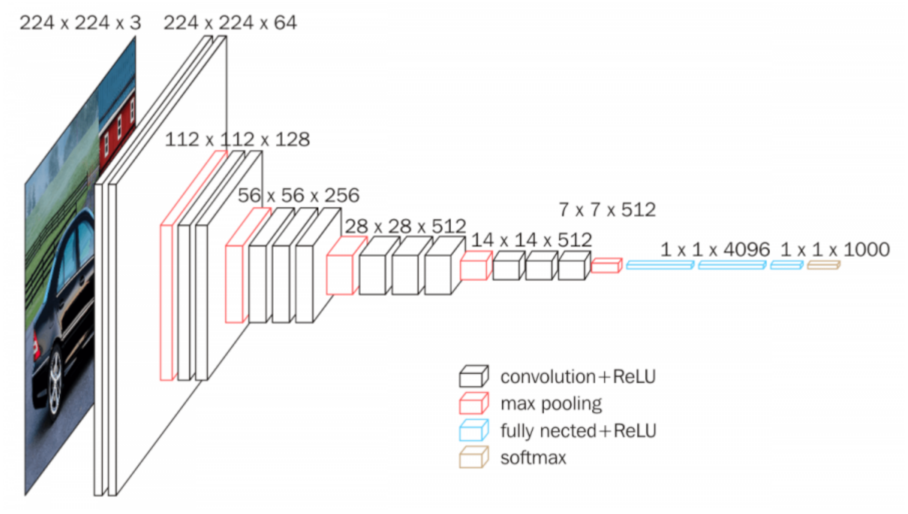


Figure 2.3: VGG16 architecture taken from [Has].

layers and Fig. 2.4 shows on the left-hand-side VGG19 with nineteen layers. In their convolutional layers, they used a very small filter size of 3×3 and a Rectified Linear Unit (ReLU) activation function. They use five convolutional blocks, whereby a block consists of several convolutional layers. To deal with the dimensions, they use the function *max pooling* between the convolutional blocks to reduce the dimensions. In the end, there is another block of fully connected layers to summarize the information. VGG16 and VGG19 belong to the most famous architectures of deep neural networks achieving a 0.9 top-5 accuracy (i.e. the label is in the top five predictions) on ImageNet [Ker]. Due to its simple structure and quite a manageable layer size, the pre-trained VGG16 is used as an example in this thesis to evaluate explanation methods.

ResNet. The *Residual Network* (ResNet) was introduced by He, Zhang, Ren, and Sun. In their paper "Deep Residual Learning for Image Recognition", they introduced a residual function to train deep neural networks with lower complexity [HZRS16]. The characteristic of residual networks is that they skip layer connections. For example, Fig. 2.4 shows on the right-hand-side a residual network with skip connections whereas in the middle there is a plain neural network, i.e. the layer are all run through there little by little. However, this increases the runtime and the number of parameters enormously, which can be overcome by using residual blocks.

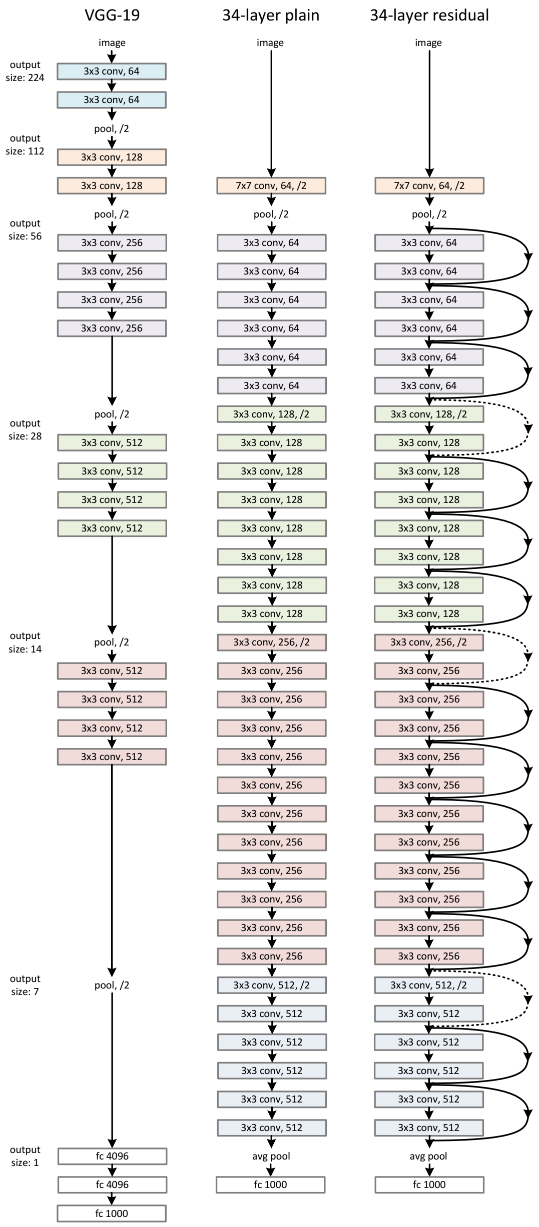


Figure 2.4: Example CNN architectures for image recognition taken from [HZRS16]. From left to right: the VGG-19 model [SZ15], a plain network with 34 parameter layers, and a residual network with 34 parameter layers.

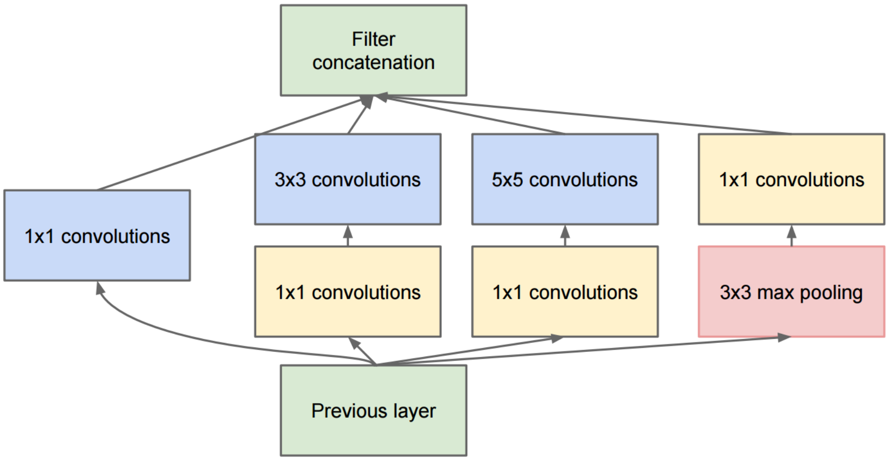


Figure 2.5: Inception layer taken from [Sze+16].

InceptionNet. The architecture *Inception* was introduced by Szegedy, Liu, Jia, Sermanet, Reed, Anguelov, Erhan, Vanhoucke, and Rabinovich and called *GoogLeNet* or Inception V1, respectively. In their paper "Going Deeper with Convolutions", they define a module that allows running multiple filters with different dimensions simultaneously. The idea is that it is not necessary to choose one particular filter size, but that all of them can be used. This facilitates the extraction of multi-level features. Such a module can be seen in Fig. 2.5. The output of a previous layer is passed on several 1×1 convolutional layers to save parameters by reducing the depth of the output and a 3×3 max-pooling layer to adjust the dimension for the different convolutional layers [Sze+16]. GoogLeNet consists of 22 layers whereas a part of these layers consists of a total of nine Inception modules. The later version provided by Keras is Inception V3 from [Sze+16].

MobileNet. The architecture *MobileNet* was introduced by Howard, Zhu, Chen, Kalenichenko, Wang, Weyand, Andreetto, and Adam. In their paper "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", they present *depthwise separable convolutions* to build lightweight neural networks [How+17], i.e. less computation power and thus also executable on mobile phones. Fig. 2.6 shows the components to build a depthwise separable convolutional filter. It is basically a factorization of a standard filter (Fig. 2.6 (a)) into a depthwise convolutional filter (Fig. 2.6 (b)) and a pointwise convolutional filter (Fig. 2.6 (c)). Although depthwise

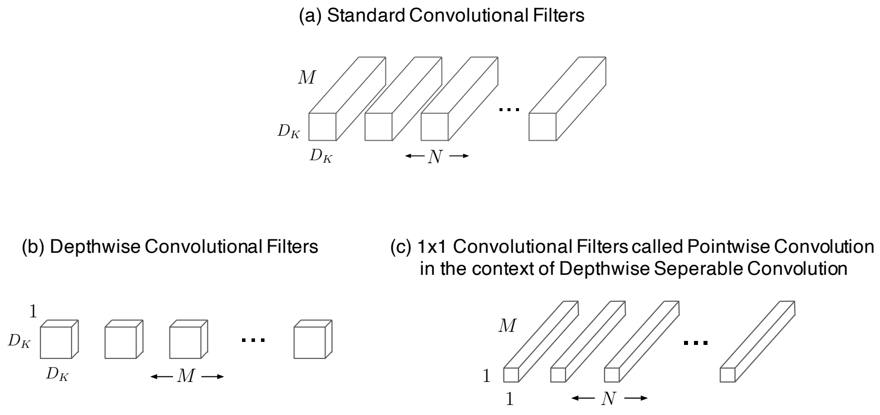

(a) Standard Convolutional Filters
M
D_K
D_K
$\leftarrow N \rightarrow$
(b) Depthwise Convolutional Filters
1
D_K
D_K
$\leftarrow M \rightarrow$
(c) 1x1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Seperable Convolution
M
1
1
$\leftarrow N \rightarrow$

Figure 2.6: Components to build a depthwise separable filter taken from [How+17].

convolution needs less computation power than standard convolution, it only filters the single channels and does not merge them. Therefore, pointwise convolution is used to bring the features of the individual channels together again. This combination is called depthwise separable convolution.

2.2 Explanation Approach

There are three different scenarios where we can start to explain a neural network. The first scenario is before building a model. Before building a model, the underlying data can be analyzed to reveal understandable structures in the data that are then processed from a neural network. By understanding the data, it is possible to derive explanations for the output. For example, Gisolfi and Dubrawski introduced a bounding box algorithm to extract simpler structures from the data. The second scenario arises while building a new model. Here, we can use rule-based [GHMS92], case-based [LC99], or monotonicity-based [AW93] approaches to build a neural network to understand the model afterward. For example, Wu, Hughes, Parbhoo, Zazzi, Roth, and Doshi-Velez [Wu+18] regularize deep models by training deep time-series models which are modeled by decision trees. These two scenarios define interpretable models. The last scenario is after building a model. Here a neural network is given. Due to the black box problem with neural networks, there are several approaches to unveil its behavior. There are attempts to approximate the neural network with an interpretable model [LL17, RSG16], to evaluate the output with sensitivity analysis [SZ19, KDS17], or investigate the hidden layers [CEP20, KKMK20]. In this thesis, we will focus on

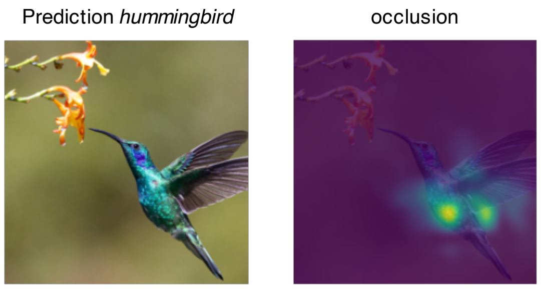


Figure 2.7: Example of the explanation method Deconvnet on VGG16.

the last scenario of a given neural network which we want to explain. Thus, in the following, we will present some approaches that occur in this thesis.

Perturbation-Based Approach

Perturbation-based approaches examine a model's output by perturbing specific input features. These explanation methods use techniques like masking, blurring, or noise to perturb an input to analyze the feedback regarding the output. The idea behind this is that in altering input features that contribute maximally to an output, the prediction of that output would decrease significantly. The different outputs can then be compared to draw conclusions [SM19].

Zeiler and Fergus firstly introduced *Deconvnet* a method to perturb visualizations [ZF14]. They alternately cover areas of an image with a gray square. The absence of information is simulated by occluding regions of the image. The gray square is gradually sliding over the entire image. The probability of the correct class is then visualized depending on the position of the occluded square. This identifies important regions in the images. However, depending on the size, color, and sliding of the square, the output may be affected. Fig. 2.7 shows a visualization of the important pixels with Deconvnet on VGG16 which predicted the correct class with 1.00 accuracy.

Therefore, Ribeiro, Singh, and Guestrin take a different approach to perturb images. With their method *LIME* (Local Interpretable Model-agnostic Explanations), they divide an image into *super-pixels*. A super-pixel is defined as a group of pixels that has specific properties such as color intensity. Then, they perturb these super-pixels one by one and predict the class with the perturbed image. In this way, the most important areas of the input for the decision can be detected and visualized. Fig. 2.8 shows a visualization of the important super-pixels with LIME on VGG16 which predicted the correct class with 1.00 accuracy.

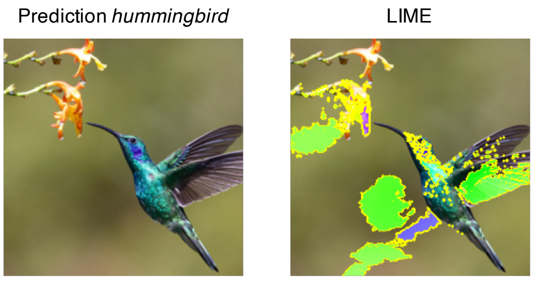


Figure 2.8: Example of the explanation method LIME on VGG16.

Backpropagation-Based Approach

Backpropagation is a method used to learn neural networks. Through backpropagation, the error of the loss function is minimized. Inspired by this, backpropagation-based approaches calculate an heat-map based on the output and the calculated weights such as *Guided Backpropagation* [SDBR15], *DeconvNet* [NHH15] and *DeepLIFT* [SGK17]. However, these approaches do not distinguish between different prediction classes.

Class Activation Mapping (CAM) removes the fully connected layers and replaces them with a global average pooling. Then, it retrains the neural network and outputs the probability for each class [Zho+16]. This modification enables the computation of a heat-map for each class. *GradCAM* (Gradient-weighted Class Activation Mapping) is an extension of CAM without modification of the neural network. It computes the gradients of the feature maps from the convolutional layers. Then, it gets a weighted heat-map by computing a global average pooling of the gradients [Sel+16]. *GradCAM++* further extends GradCAM by only considering positive gradients. The idea is that only pixels that have a positive impact on the predicted class should be visualized [CSHB18]. Fig. 2.9 shows a visualization of the important pixels with GradCAM and GradCAM++ on VGG16 which predicted the correct class with 1.00 accuracy.

In this thesis, we will apply *Layerwise Relevance Propagation* (LRP) which is a conservative method. It computes the relevance of each input neuron to each output neuron with backpropagation while preserving the activation values [Bac+15, Mon+19, MSM18]. Following [NKHF21] and the definitions in Section 2.1, the LRP

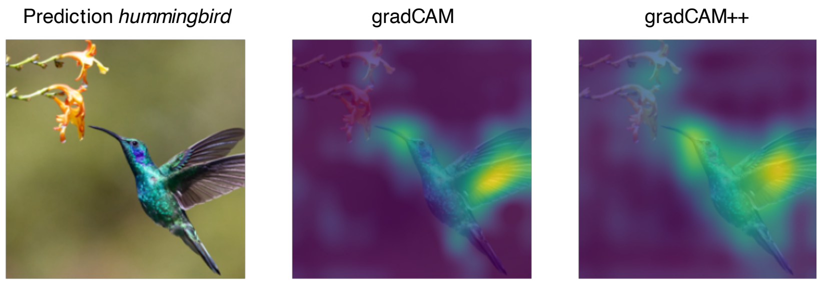


Figure 2.9: Example of explanation methods GradCAM and GradCAM++ on VGG16.

algorithm is defined as

$$R^z_{(k,i)} = \sum_{j \in \Pi_{k+1}} \frac{z^k_i w^{k,k+1}_{ij}}{\left(\sum_{i \in \Pi_k} z^k_i w^{k,k+1}_{ij} \right) + b^k_j} R^z_{(k+1,j)}.$$

3 TransPer

Neural networks are a popular tool in e-commerce, in particular for product recommendations. To build reliable recommender systems, it is crucial to understand how exactly recommendations come about. Unfortunately, neural networks work as black boxes that do not provide explanations of how the recommendations are made.

In this chapter, we present *TransPer*, an explanatory framework for neural networks. It uses novel, explanation measures based on *Layer-Wise Relevance Propagation* and can handle heterogeneous data and complex neural network architectures, such as combinations of multiple neural networks into one larger architecture. We apply and evaluate our framework on two real-world online shops. We show that the explanations provided by *TransPer* help (i) understand prediction quality, (ii) find new ideas on how to improve the neural network, (iii) help the online shops understand their customers, and (iv) meet legal requirements such as the ones mandated by GDPR. In particular, we aim to answer the following research questions with our approach.

Research Question 1. Hypothesis: Quantifying explanations based on the relevance of the input features facilitates the evaluation of not only the input data but also the neural network.

1.1 What parameters are relevant for understanding explanation quality?

1.2 Which implications can be derived with these relevancies?

1.3 Can these relevancies be used to improve the neural network (and if so, how)?

This chapter extends work initiated as part of a project in collaboration with the organization econda and is based on joint work with Franz Krause, Daniel Hagenmayer, and Michael Färber [NKHF21].

3.1 Introduction

The breakthrough with neural networks as a pattern recognition technique has led its way into many industry sectors. Especially in e-commerce, it can be used as a recommender system for advanced searches [LM20], personalization of shopping experiences and direct marketing [Par00], or advanced sales forecasting and predictions [LMS18]. Improving the predictions and the usefulness of those recommenders can increase sales and customer satisfaction. Additionally, there is increasing legal pressure in favor of privacy and data protection. For example, the General Data Protection Regulation [PE16] (GDPR) states that data subjects should be enabled to check the collection, processing, or use of their data. Thus, businesses may be legally required to make their recommender systems transparent.

Multilayer Perceptrons (MLP) have been applied in recommender systems learning feature representations as an extension to collaborative filtering [KTL20]. In combination with convolutional layers, they are applied to generate fashion outfits for e-commerce or to personalize outfit recommendations based on learned embeddings in Convolutional Neural Networks (CNN) [BHZC20, Che+19]. Recurrent Neural Networks (RNN) have shown success in modeling sequential data and have been used for personalized product recommendations based on the purchase patterns of customers [ND19], learning embeddings of fashion items [LCZL17] and modeling user behavior to predict clicks [BMRS16].

However, neural networks are black-box models, i.e., the predictions can not be explained. To tackle this, it is beneficial to make them more transparent and therefore, more human-understandable. Typically, the Gradient-based Sensitivity Analysis [RHW86] is used to explain the predictions of neural networks. By optimizing the gradient ascent in the input space, it is possible to determine which inputs lead to an increase or decrease of the prediction score when changed [SVZ14, STY17]. Although applications based on this method enable a statement regarding the positive or negative influence of an input on a prediction, they do not reveal a quantitative decision-relevant input score such as Guided Backpropagation [SDBR15], DeconvNet [NHH15], or DeepLIFT [SGK17]. These algorithms use the trained weights and activations within the forward pass to propagate the output back to the input. This way, it is possible to determine which features in an input vector contribute to the classification and what extent. Exploiting this, ObAlEx [NOF21] is an explanation quality metric which measures to what extent the classified object is aligned to the

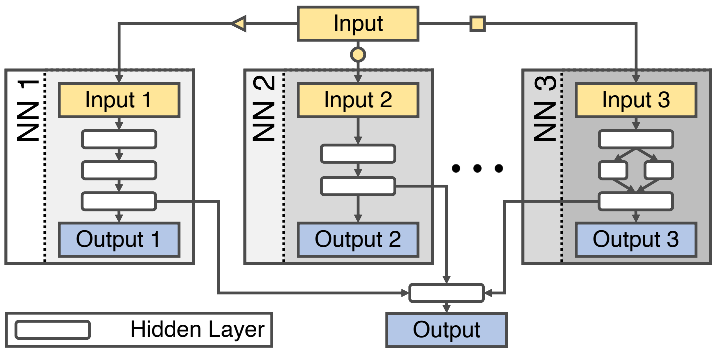


Figure 3.1: Model of a neural network recommender with different input data types

mentioned explanations. Nonetheless, all these methods are solely applied to CNNs with image data where single pixels are then highlighted. Another back-propagating algorithm is the Layer-Wise Relevance Propagation (LRP) that has already been successfully used in interaction with MLPs and CNNs [ACÖG18, Bac+15, Mon+19]. LRP computes the relevance of each input neuron to the output by performing a value-preserving backpropagation of the output. Furthermore, this method is even applicable on RNNs with sequential data [Bha18, MSM18] which often occurs in processing customer profiles in e-commerce.

Although there are already some approaches in this field, it is now the case, due to the data basis, that not only certain types of data are fed into a neural network. Considering an online shop for clothes, both the images of the clothes or their categories and the click history of the customers can be fed into the neural network. Thus, this neural network would handle different types of data. Since the above approaches only consider one type of neural network and therefore one datatype, we need an explanation approach that can handle different types of data as in Fig. 3.1.

Contribution. Our contribution is threefold. First, we provide an explanation framework called TransPer[1] for e-commerce businesses in online shopping (e.g., for product recommendation) to provide transparency to the neural networks used. Based on a custom implementation of *Layer-Wise Relevance Propagation*, our approach can not only handle individual neural networks types, but also more complex architectures that contain multiple neural subnetworks, such as shown in Fig. 3.1. This is required in the presence of highly heterogeneous input data (e.g., product images, chronological shopping interactions, personal information) where different neural network types are necessary (e.g., CNN, RNN, MLP). We not only take into

[1] We provide the source code online at https://github.com/Krusinaldo9/TransPer.

account the relevance of the activations of the neurons, but also the bias. This has not been considered in depth in the literature. Second, we define quantity measures to evaluate the helpfulness of these explanations. The individuality measure can be used to determine those parts of the input that are particularly relevant for the decision. The certainty measure quantifies how certain the system is about its prediction. The diversity measure states whether there are clear top predictions. Third, we evaluate our approach in real-world scenarios. To this end, we used data from two real-world online shops provided by our partner *econda*, an e-commerce solution provider. We show that TRANSPER helps in (i) understanding the prediction quality, (ii) finding ideas to improve the neural network, and (iii) understanding the customer base. Thus, TRANSPER brings *trans*parency to *per*sonally individualized automated neural networks and provides new knowledge about customer behavior. We believe that this helps to fulfill GDPR requirements.

The remainder of this chapter is structured as follows. After introducing preliminary definitions and concepts in Section 3.2, we go on to describe the problem setting and formally define an online shop in Section 3.3, to introduce our quantity measures in Section 3.4. We evaluate our approach based on a real-world scenario in Section 3.5 before ending with some concluding remarks.

3.2 Preliminaries

In this section, we present the fundamentals for the application of our approach. To begin with, we consider a trained neural network with $K \in \mathbb{N}$ layers as shown on the left-hand side of Fig. 3.2. We refer to Π_k as the set of all neurons in the k-th layer, σ as a nonlinear monotonously increasing activation function, z_i^k as the activation of the i-th neuron in the k-th layer, $w_{ij}^{k,k+1}$ as the weight between the neurons z_i^k and z_j^{k+1}, and b_j^k as the bias term w.r.t. z_j^{k+1}. Assuming that we know the activations in Π_k, the activations in Π_{k+1} can be determined via forward pass as follows:

$$z_j^{k+1} = \sigma\left(\left(\sum_{i \in \Pi_k} z_i^k w_{ij}^{k,k+1}\right) + b_j^k\right) \tag{3.1}$$

For non-connected neurons z_i^k and z_j^{k+1} we assume $w_{ij}^{k,k+1} = 0$. If a network has no bias, then $b_j^k = 0$.

Layer-Wise Relevance Propagation is a method that represents a backward analysis method [Bac+15]. Knowing the activations z_j^{k+1} in layer $k+1$, we can determine to what extent the neurons in Π_k and the biases b_j^k have contributed, or how relevant they were. The idea behind the standard implementation of the LRP algorithm can

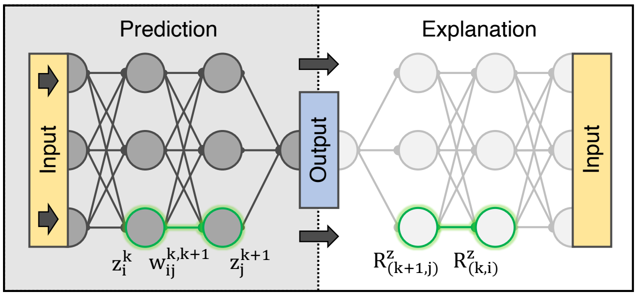


Figure 3.2: Exemplary run of LRP. The left-hand side shows the calculation of neuron activations in the forward pass. These activations are then part of the calculation of its relevancies in the backward analysis depicted on the right-hand side.

be found on the right-hand side of Fig. 3.2 and is defined as

$$R^z_{(k,i)} = \sum_{j \in \Pi_{k+1}} \frac{z_i^k \, w_{ij}^{k,k+1}}{\left(\sum_{i \in \Pi_k} z_i^k \, w_{ij}^{k,k+1}\right) + b_j^k} R^z_{(k+1,j)}, \tag{3.2}$$

$$R^b_{(k,j)} = \frac{b_j^k}{\left(\sum_{i \in \Pi_k} z_i^k \, w_{ij}^{k,k+1}\right) + b_j^k} R^z_{(k+1,j)}. \tag{3.3}$$

For a layer $k+1$, we assume for each neuron j that a relevance can be assigned in the form of a real-valued number $R^z_{(k+1,j)}$. Using Equation 3.2, we obtain the relevance, i.e., quantitative contribution, of the i-th neuron in the k-th layer to the overall relevance of layer $k+1$. Furthermore, Equation 3.3 provides the relevance of the bias b_j^k of the j-th neuron in layer $k+1$.

In certain applications, customized variations of the standard LRP algorithm presented above can be considered to increase the performance. In particular, regarding the explainability of CNNs, it has been found that adapted LRP methods lead to better results than the standard LRP method [ACÖG18, Bac+15, Mon+19]. These are characterized, e.g., by the use of tuning parameters or penalty terms for negative neuron activations. Regarding RNNs, however, hardly any results exist concerning the use of such variations. Therefore, with the use cases in Section 3.5, we provide results of a test study comparing well-known customizations with the standard method.

3.3 Formal Model of an Online Shop

In this section, we define an online shop regarding a suitable neural network that can handle specific characteristics. Especially, we include heterogeneous input data such as interest in products or interactions with products which additionally can have different input lengths. To generalize our definition, we consider a neural network consisting of several neural subnetworks to cover different cases as can be seen in Fig. 3.1. Considering all this, we define our online shop as follows.

Definition (Online Shop Model)**.** *We define an online shop T as a tuple*

$$T = (C, P, (P^*, \Phi), \Lambda, \Lambda^*, S, (\Omega_c)_{c\in C}, (\omega_c)_{c\in C}, (f_c)_{c\in C})$$

with the following entries:

a) We denote C as the finite set of all customers of the shop.

b) Let P be the finite set of all products that the shop offers.

c) Then, let P^* be a subset of P or P itself, i.e., $P^* \subseteq P$, and Φ denotes the real-valued output space $[0,1]^{|P^*|}$.

d) We denote Λ as the set of information types that the shop T can have about one of its customers $c \in C$ and assume that this amount is finite.

e) We define Λ^* as a finite set of disjoint subsets $\Lambda_1, ..., \Lambda_n$ of Λ which corresponds to neural networks $S = \{s_1, \cdots, s_n\}$.

f) For a customer $c \in C$ we define an associated real-valued input space

$$\Omega_c = \mathbb{R}^{m_1(c)} \times ... \times \mathbb{R}^{m_n(c)}$$

with the mappings $m_i : C \longrightarrow \mathbb{N}$ for $i \in \{1, .., n\}$ with respect to s_i.

g) Considering a particular customer $c \in C$, we define his input as $\omega_c \in \Omega_c$.

h) For a customer $c \in C$, we also define the mapping $f_c : \Omega_c \longrightarrow \Phi$ where $f_c(x)$ is the recommender's output vector for an input $x \in \Omega_c$.

Assume we have an online shop T with customers C. The online shop has a catalog of offered products P. Though, not all products are predicted for example only seasonally available ones or most purchased ones in the last week denoted by P^*. These are used as output space Φ in the neural network, i.e., if $\Phi(p) > \Phi(p')$ then product p is recommended. Now, consider the types of information Λ the online shop can have about their customers such as already purchased products, interactions, or ratings. As mentioned in Section 3.1, certain network types are more suitable for specific data types. Therefore, this information is then classified into disjoint information types, such as sequential data Λ_1, graphical data Λ_2, etc., and summarized in Λ^*. So, if an online shop T has heterogeneous user data, Fig. 3.1 would consist of neural subnetworks $s_1, \cdots, s_n$. With homogeneous data, we would have a special case of the previous one. Hence, we have:

1. $\Lambda = \Lambda'$ and $\Lambda^* = \{\Lambda'_1, ..., \Lambda'_n\}$. (heterogeneous data) (3.4)
2. $\Lambda = \Lambda'_i$ for some i and $\Lambda^* = \{\Lambda'_i\}$. (homogeneous data) (3.5)

The different Λ'_i can have different input lengths depending on the sequence length of the interactions or the size of the images. So, we use the mappings m_i to deal with it and summarize them in Ω. Thus, for a customer $c \in C$, we obtain the neural network's output vector $y = f_c(\omega_c)$.

Considering the different data types, the online shop has three possibilities to define a suitable neural network: (i) The online shop uses n different data types, i.e., heterogeneous data, and needs n different neural subnetworks. An overall decision is obtained by concatenating the hidden layers at a suitable position, see Fig. 3.1. (ii) Second, the online shop decides to just use one data class, i.e., homogeneous data, and therefore has just one neural subnetwork in Fig. 3.1. However, important information can be lost from the other data classes. (iii) Third, it is possible to define suitable neural subnetworks for $n > 1$ data classes, train them separately and then save their weights. These n trained neural subnetworks can be concatenated and trained again with the entire data, using the already trained weights and biases as initial values. This approach is therefore a combination of the two mentioned possibilities above. Thus, $n + 1$ neural subnetworks are obtained in total, with one resulting from the concatenation of the n individual neural subnetworks. The output vector then depends on whether one uses the concatenated network s_{n+1} or one of the neural subnetworks $s_1, ..., s_n$. This third possibility will be relevant for our use case.

3.4 Explanation Approach

The goal of our approach is to evaluate the explanation of product recommendations of a shop-adapted neural network to better understand the decision. Given input from a user of an online shop and a trained neural network as a recommender, TransPer performs a backward analysis based on an individual prediction. In this way, it can be explained to what extent components of the trained network or certain inputs were relevant. This process can be seen in Fig. 3.3. In the following, we will (i) describe how these explanations can be gained with LRP, (ii) specify how to analyze the input

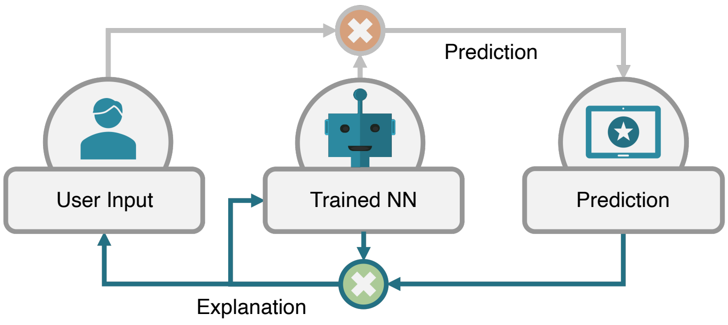


Figure 3.3: TransPer Overview.

with the Leave-One-Out method, and (iii) define quantity measures to evaluate the explanations.

3.4.1 Explanation via Layer-Wise Relevance Propagation

Following the notation of Section 3.2 and Definition 3.3, we assume that $K \in \mathbb{N}$ denotes the number of layers in the neural network, i.e., the first layer is the input layer and the K-th layer is the output layer. Furthermore, for $k \in \{1, ..., K\}$ let $|\Pi_k| = I_k \in \mathbb{N}$ be the number of neurons in the k-th layer, i.e., I_1 describes the number of input neurons and I_K the number of output neurons. Indeed, in the context of classifiers, each neuron of the output layer represents one element of the target set. For example, for an input x, the neuron (K, i^*) with the highest prediction score $f(x)_{i^*}$ as output is the actual recommendation. In this context, it is then of interest to find out to what extent the neurons of the lower layers contributed to the decision $f(x)_{i^*}$. For our approach, we define the initial relevance vector $R^z_{(K,\cdot)} := (R^z_{(K,i)})_{i\in\{1,...,I_K\}}$ with

$$R^z_{(K,i)} = \begin{cases} f(x)_{i^*} & \text{if } i = i^* \\ 0 & \text{otherwise} \end{cases}$$

which can be used to iteratively compute the relevance for layers $K - 1, ..., 1$ using Equation 3.2 and Equation 3.3. Finally, we obtain $R^z_{(1,\cdot)}$ as the input layer's relevance vector and can thus determine to what extent an input neuron is decision-relevant (see Fig. 3.2). Note that a negative relevance in an input neuron diminishes the prediction i^* whereas a positive relevance underpins it. In contrast to most LRP approaches, we also consider the relevance of the bias $R^b_{(k,j)}$ of the j-th neuron of the $(k+1)$-th layer. Our LRP method is characterized as follows:

$$\begin{aligned}
\sum_{i\in\Pi_k} R^z_{(k,i)} + \sum_{j\in\Pi_{k+1}} R^b_{(k,j)} &= \sum_{i\in\Pi_k}\sum_{j\in\Pi_{k+1}} \frac{z_i^k w_{ij}^{k,k+1}}{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k} R^z_{(k+1,j)} \\
&\quad + \sum_{j\in\Pi_{k+1}} \frac{b_j^k}{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k} R^z_{(k+1,j)} \\
&= \sum_{j\in\Pi_{k+1}} \frac{\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}}{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k} R^z_{(k+1,j)} \\
&\quad + \sum_{j\in\Pi_{k+1}} \frac{b_j^k}{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k} R^z_{(k+1,j)} \\
&= \sum_{j\in\Pi_{k+1}} \frac{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k}{\left(\sum_{i\in\Pi_k} z_i^k w_{ij}^{k,k+1}\right)+b_j^k} R^z_{(k+1,j)} \\
&= \sum_{j\in\Pi_{k+1}} R^z_{(k+1,j)}.
\end{aligned}$$

As $f(x)_{i^*} = \sum_{j\in\Pi_K} R^z_{(K,j)}$ is satisfied by assumption, we obtain

$$\begin{aligned}
f(x)_{i^*} &= \sum_{i\in\Pi_{K-1}} R^z_{(K-1,i)} + \sum_{j\in\Pi_K} R^b_{(K-1,j)} \\
&= \sum_{i\in\Pi_{K-2}} R^z_{(K-2,i)} + \sum_{j\in\Pi_{K-1}} R^b_{(K-2,j)} + \sum_{j\in\Pi_K} R^b_{(K-1,j)} \\
&= \cdots
\end{aligned}$$

$$= \underbrace{\sum_{i\in\Pi_1} R^z_{(1,i)}}_{=:R^z} + \underbrace{\sum_{k=1}^{K-1}\sum_{j\in\Pi_{k+1}} R^b_{(k,j)}}_{=:R^b}, \tag{3.6}$$

i.e., the sum of the final relevancies R^z and R^b equals the original output score. By comparing the two summands in Equation 3.6, the LRP algorithm also provides a method to find out how much relevance R^z, R^b can be assigned to the input neurons and the trained bias, respectively.

3.4.2 Input Analysis with Leave-One-Out Method

In this section, we want to find out why well-functioning recommenders work and provide new insights into the customers' shopping behavior. Additionally, we want to know why an insufficiently functioning recommender delivers meaningless predictions. Therefore, we need to further analyze the explanations gained from LRP regarding their helpfulness, i.e., the impact of input on the prediction. Using the Leave-One-Out method [YLLZ10], we evaluate the input relating to the explanations. By consistently leaving one product out by setting its input value to zero, we can observe its effect on the predictions and explanations, see Fig. 3.4. Assuming a trained neural network, we perform the following steps:

(i) We start with a particular customer and the associated input x which is mapped to an output vector y via the trained network.

(ii) According to Equation 3.6, for a given output neuron y_{i^*} with $i^* \in \{1,..,I_K\}$ (e.g., the one with the highest prediction score), we compute the associated input relevancies $(R^z_{(1,j)})_{j\in\{1,..,I_1\}}$ and the overall relevance of the bias R^b.

Thus, we consider the set of relevancies $R := \{R^b\} \cup \{R^z_{(1,j)} : 1 \leq j \leq I_1\}$.

(iii) For a salient subset of the relevancies $R^* \subset R$ (e.g., the inputs with the highest/lowest relevancies), we set the associated input neurons (marked red in Fig. 3.4) in x to 0 and obtain the adapted input vector x^*.

(iv) As in Step (i), we map the input x^* to the corresponding output y^* via the same trained network and obtain the test output y^*.

Thus, with steps (i)-(iv), we obtain the input vectors x and x^*, the output vectors y and y^*, and the set of relevancies R. They are used in Section 3.4.3 to enable the explainability of neural network predictions according to the online shop in Definition 3.3.

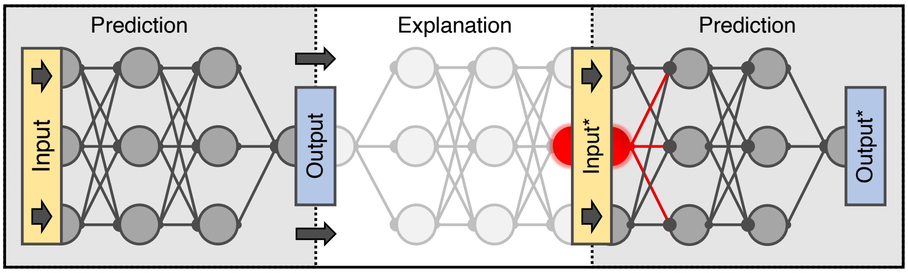


Figure 3.4: Selection and analysis of the most relevant inputs via LRP

3.4.3 Explanation Quantity Measures

Methods such as A/B testing exist to test the performance of a recommender system [GV16, CL17]. They aim at evaluating the predictions trained on a fixed group of customers with new test customers. Ideally, positive feedback on the training process is obtained. However, the results can be unsatisfactory as well. In both cases, it is of interest to know how the predictions come about and how certain inputs influence them specifically. Using Equation 3.6 and the definitions

$$R^z_+ := \sum_{i \in \Pi_1} \max\{0, R^z_{(1,i)}\}, \quad R^z_- := \sum_{i \in \Pi_1} \min\{0, R^z_{(1,i)}\},$$

we obtain the network's top prediction within the setting of Definition 3.3

$$y_{i^*} := f_c(\pi_c)_{i^*} = R^z + R^b = R^z_+ + R^z_- + R^b. \tag{3.7}$$

In the following we consider two disjoint subsets $C_1, C_2 \subset C$. C_1 represents a set of customers where the inconsistencies to be analysed occur. In contrast, this is not the case for customers from C_2. With Equation 3.7, it is then possible to define measures that can be used to analyse such irregularities in specific test cases. W.l.o.g we always assume for the output value $y_{i^*} > 0$. Based on these considerations, we define three measures to quantify the relevance of the input.

Individuality Measure.

Definition (Individuality Measure). $\sigma_T : C \longrightarrow \mathbb{R}$ *with*

$$\sigma_T(c) := \frac{R^z}{R^z + R^b} = \frac{R^z}{y_{i^*}}.$$

The individuality measure can be used to determine to what extent the input was relevant for the decision. Via Equation 3.7, we obtain $1 = R^z/y_{i^*} + R^b/y_{i^*}$ and define that a prediction y_{i^*} is *maximally individual*, if $\sigma_T(c) = 1$ holds. In contrast, y_{i^*} is considered to be *minimally individual*, if $\sigma_T(c) = 0$ holds. In this case only the bias was relevant. For $\sigma_T(c) \in (0, 1)$ we generally have $R^z, R^b > 0$, so both of these components contribute positively to y_{i^*}. If R^z or R_b are negative, this component argues against prediction y_{i^*} and we either have $\sigma_T(c) \in (-\infty, 0)$ or $\sigma_T(c) \in (1, \infty)$. Note that due to $y_{i^*} > 0$ it can not occur that R^z and R_b are negative.

With σ_T it is for example possible to attribute inconsistencies to overly homogeneous training data. Consider a shop offering men's and women's products. Let men be C_1 and women be C_2. If the training data is largely assigned to men, women could often get men's products suggested because the recommender's bias was trained on men. Then, for $c_1 \in C_1$ and $c_2 \in C_2$, the following would apply: $|1 - \sigma_T(c_1)| < |1 - \sigma_T(c_2)|$.

Certainty Measure.

Definition (Certainty Measure). $\nu_T : C \longrightarrow (0, 1]$ *with*

$$\nu_T(c) := \begin{cases} R^z/R^z_+, & \mathit{if}\, R^z > 0 \\ R^z/R^z_-, & \mathit{if}\, R^z < 0. \end{cases}$$

The certainty measure can be used to make a quantitative statement about the deviation of the individual relevancies from the overall relevance. Considering definitions of R^z_+, R^z_-, and Equation 3.7, we have $R^z_+ \in [R^z, \infty)$ and $R^z_- \in (-\infty, R^z]$. Depending on the sign of R^z, one can determine whether the input neurons as a whole had a positive or negative relevance for the decision made. We restrict ourselves to the case of $R^z > 0$. However, the results apply to $R^z < 0$, respectively. Thus, we can deduce that a value of $\nu_t(c) = 1$ means that no negative relevancies were assigned to the input neurons. A value close to zero, on the other hand, indicates a strong dispersion of the relevancies.

Diversity Measure.

Definition (Diversity Measure). $\zeta_T, \zeta_T^+, \zeta_T^- : C \longrightarrow [0, \infty)$ *with*

$$\zeta_T(c) := \max_{r \in \mathcal{R}} \left| \frac{r - \mu^r_{\mathcal{R}}}{\mu^r_{\mathcal{R}}} \right| \quad \mathit{and} \quad \mu^r_{\mathcal{R}} := \frac{1}{|\mathcal{R}| - 1} \sum_{r' \in \mathcal{R} \setminus \{r\}} r',$$

$$\zeta_T^+(c) := \max_{r \in \mathcal{R}_+} \left| \frac{r - \mu^r_{\mathcal{R}_+}}{\mu^r_{\mathcal{R}_+}} \right| \quad \mathit{and} \quad \mu^r_{\mathcal{R}_+} := \frac{1}{|\mathcal{R}_+| - 1} \sum_{r' \in \mathcal{R}_+ \setminus \{r\}} r',$$

$$\zeta_T^-(c) := \max_{r \in \mathcal{R}_-} \left| \frac{r - \mu^r_{\mathcal{R}_-}}{\mu^r_{\mathcal{R}_-}} \right| \quad \textit{and} \quad \mu^r_{\mathcal{R}_-} := \frac{1}{|\mathcal{R}_-| - 1} \sum_{r' \in \mathcal{R}_- \setminus \{r\}} r'$$

for a customer $c \in C$ and top prediction y_{i^*}. We additionally introduce the set of input relevancies $\mathcal{R} := R^z_{(1,\cdot)}$, which we divide as follows:

$$\mathcal{R}_0 := \{r \in \mathcal{R} : r = 0\},\ \mathcal{R}_+ := \{r \in \mathcal{R} : r > 0\},\ \textit{and}\ \mathcal{R}_- := \{r \in \mathcal{R} : r < 0\}.$$

The diversity measure finds outliers within certain input relevancies. For example, considering $r \in \mathcal{R}$, then $(r - \mu^r_{\mathcal{R}})/\mu^r_{\mathcal{R}}$ is the proportional deviation between the values in $\mathcal{R}$ except for r. For $r \in \mathcal{R}_+$ or $r \in \mathcal{R}_-$ one proceeds analogously. Note that the calculation of diversity measures does not apply to empty sets $\mathcal{R}, \mathcal{R}_+$, and $\mathcal{R}_-$, respectively. Furthermore, the zero is always obtained for one-element sets. For two customers c_1, c_2 with $\zeta_T^+(c_1) \ll \zeta_T^+(c_2)$, we can thus state that the prediction for c_2 depends more on a single input neuron than the prediction for c_1.

3.5 Evaluation

In this section, we demonstrate the benefits and application of our approach in three use cases. First, our explanation approach can help in understanding fluctuations in the recommender's quality. Second, TransPer can help in finding ideas on how to improve the recommender. Third, our contribution can help to improve the understanding of the customer base. In the course of this research, we kindly received permission from the e-commerce service provider econda [eco] and two of its partner companies to use their customer data. These partner companies are a jewelry shop and an interior design shop.

3.5.1 Evaluation Setting

At this point, we show that both online shops fit the formal model from Definition 3.3 and are thus applicable to the TransPer framework. We assume that T^1 is the jewelry shop and T^2 is the interior design shop. As shortly mentioned in Section 3.3, the neural network econda uses for T^1 and T^2 comply with the third neural network type with three neural subnetworks s_1, s_2, s_3 in Fig. 3.1.

Online Shop Models.. We now illustrate how the shops satisfy Definition 3.3:

a) Both shops provide anonymized information about a variety of their customers $\widetilde{C}^1 \subseteq C^1, \widetilde{C}^2 \subseteq C^2$, for example shopping history,

b) and their offered products P^1, P^2.

c) The targets P^*, in our use case a subset of selected products of the offered products, define the real output space Φ^1 and Φ^2, respectively.

d) The available customer information types are based on the information sets Λ^1 and Λ^2, respectively.

e) The information from Λ^1 (Λ^2) is classified according to its characteristic properties. In our case, the disjoint subsets are the same for both shops, i.e., $\Lambda^* = \Lambda^{1*} = \Lambda^{2*}$. Especially, T^1 and T^2 have three disjunctive information types, i.e., $|\Lambda^*| = 3$, which result in three neural subnetworks s_1, s_2, s_3.

f) According to Λ^*, any customer c has therefore the associated input space denoted by $\Omega_c = \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \times \mathbb{R}^{m_3(c)}$. The first two neural subnetworks s_1, s_2 have a fixed number of input neurons independent of the customer, so in a slight abuse of notation we write m_1 and m_2 instead of $m_1(c)$ and $m_2(c)$, respectively. The third subnetwork has several neurons dependent on the number of interactions of c.

g) Via preprocessing, the information about a user $c \in C$ is converted into an input $\omega_c \in \Omega_c$.

h) The function f_c represents the recommender's implicit process of decision making. Given an input ω_c, the vector $f_c(\omega_c)$ contains an entry for each product in P^* and the product with the corresponding highest prediction score is recommended.

The neural networks are trained in two steps, respectively. First, the neural subnetworks s_1, s_2, s_3 are trained independently. Based on the trained weights and biases, the subnetworks are concatenated according to Fig. 3.1 in their hidden layers and trained again to obtain the combined decision function f_c. This also means, each of the subnetworks s_1, s_2, s_3 individually fits Definition 3.3 and processes the following information types which we will further analyze in Section 3.5.3. (i) s_1 processes information regarding general interactions, whereby the input vector is an embedding of a user profile. For example, an input neuron can represent the purchase of a certain product or interest in a product category. This neural subnetwork is designed as a multi-layer perceptron. (ii) s_2 processes personal information not related to former product interactions. A multi-layer perceptron is used as well. (iii) s_3 processes the most recent customer interactions as sequences, whose lengths may be different for each customer. An action performed by a user is embedded and considered as a part of the interaction sequence. An RNN approach with Gated Recurrent Unit layers is used here.

Table 3.1: Characteristics of the data set.

Characteristics	Jewelry shop T^1	Interior design shop T^2
period of survey	December 2020	January 2021
profile stream	8	10
customers per stream	524	1004
average customer interaction	33	64

3.5.2 Evaluation Data Set

The data set used in this work consists of the online shops T^1 and T^2 as instantiations of the model from Definition 3.3. For each online shop, the corresponding recommender is provided in the form of a trained neural network. In both cases, the neural networks were trained each on the interactions of eleven consecutive days. At T^1, there were about 1000 customer profiles and at T^2 about 600 profiles. A customer profile always includes the interactions of the last 14 days. Furthermore, we receive the profile stream, which contains the user information about the customers which were previously considered as training and test data. econda updates the respective recommender at regular time intervals based on current purchasing behavior. Therefore, the data set used includes several profile streams and recommenders per online shop. In total, we use 8 (10) profile streams for T^1 (T^2). A profile stream contains on average 524 (1004) customers and per customer, we have on average 33 (64) customer interactions. All recommenders were realized in Python 3.7 with Tensorflow v2.1.0.

3.5.3 Evaluation Results

In Section 3.2, we have defined the standard LRP method. However, there are also variants of this method that outperform the standard on some architectures. To the best of our knowledge, it is not known which of these methods works best for RNNs. As a preliminary step, we, therefore, fill in this gap by evaluating the performance of the standard LRP and some of its most popular variants using our algorithm from Section 3.4.2. As a reference, we switch off each input neuron once at a time to find the neuron that is most relevant to the decision. This is the case when the change of the original prediction value is maximal by leaving out this specific input. Finally, per the LRP variant, we determine the relative frequency to detect the most relevant input neuron. Regarding the mentioned LRP methods, we first consider all possible parameter combinations for the values 0.01, 0.1, 1, 5, 10, and then choose

Table 3.2: Results of LRP-comparison for recurrent neural networks.

LRP method	Score
standard [Bac+15]	**0.9800**
epsilon [ACÖG18]	0.9560
gamma [Mon+19]	0.9080
alphabeta [MSM18]	0.7720
nonnegative [MSM18]	0.5040

the best combination. Based on the results in Table 3.2 and because the standard method achieved a hit rate of 100% in the case of MLPs, we will limit ourselves to this method. In the following, we describe three use cases that can be achieved with our explanation quantity measures defined in Section 3.4.3.

Understanding the Recommendation Quality. To tackle this, we have to examine discrepancies between prediction and input. We found one within the predictions provided by econda for the jewelry shop T^1 that could not be explained intuitively. Therefore, we apply the measures from Section 3.4.3 to obtain explanations regarding the recommender's decisions. The upper part of Fig. 3.5 shows two exemplary output layers of the neural subnetwork s_1, where $C_1, C_2 \subset C^1$ are disjoint subsets of customers C^1 of T^1. The exemplary customers were each randomly selected from 25 customers in C_1 and 29 customers in C_2, respectively. The output neurons are ranked in descending order regarding their prediction score. It can be seen that the preferred outputs for customers from C_1 are almost indistinguishable. In contrast, the scores for customers from C_2 imply clear top predictions. Considering the lower part of Fig. 3.5, we plot the residuals after setting the most relevant input neuron to zero to show the discrepancy. For customers from C_1, the discrepancy between the top prediction and the average prediction score is much smaller than for customers from C_2 because the entire curve hovers quite closely around its average. Thus, the product recommender s_1 of T^1 is not as certain about its decisions because the predictions range over a small interval. Therefore, we consider the top predictions in each case and try to gain new insights into the decision-making of the neural network via the explanation measures from Section 3.4.3. Table 3.3 shows these results including significant differences between C_1 and C_2:

(i) Comparing the results of the **individuality measure** σ_{T^1}, we can see that predictions for customers of C_1 depend more on the bias induced by the training data. Predictions for customers of C_2 are almost independent of the bias.

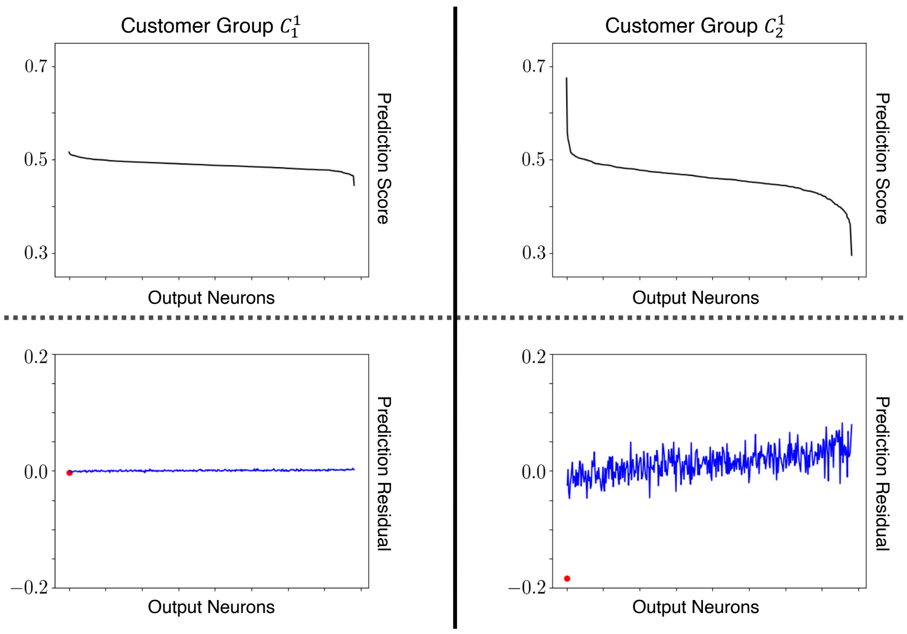


Figure 3.5: Two exemplary output layers for shop T^1. Output vectors of the NN ranked in descending order for customer groups C_1 and C_2 in the upper part including corresponding residual plots after setting the most relevant input neuron to zero in the lower part. The residual of the original top prediction is marked red.

(ii) Regarding the **certainty measure** ν_{T^1}, customers of C_1 have more contradictory input neurons with negative relevance.

(iii) Since we are interested in the positive influence of input neurons on the overall decision, we consider the **diversity measure** $\zeta^+_{T^1}$. We can see the greatest divergence between customers of the two classes C_1 and C_2. Regarding the inputs with positive relevance, customers of C_2 have an input with a relevance that is significantly greater than the other relevancies. This means that there are inputs that speak in favor of the decision making which is not the case for customers from C_1.

All three measures reveal differences between the two customer groups. The diversity measure stands out particularly prominently. The key figures listed here reflect a well explainable prediction of the recommender for customers from C_2. This means that few input neurons had the strongest influence on the prediction made which is

Table 3.3: Results of LRP-comparison for recurrent model

measure\user	C_1	C_2	C^1
σ_{T^1} (individuality)	1.2668	1.0021	1.1409
ν_{T^1} (certainty)	0.7302	0.9733	0.8804
$\zeta^+_{T^1}$ (diversity)	1.5564	143.1009	65.7103

not the case for customers from C_1. This discrepancy can also be seen very well if we switch off the input with the highest relevance and plot the residuals of the output vectors, see the lower part of Fig. 3.5. The input with the highest relevance is marked red. It has a significantly stronger influence on the prediction for customers from C_2 than C_1. For the latter, switching off this input causes almost no deviation in the predictions. Using the LRP approach and the explanatory measures, it has thus been possible to establish that the clear predictions for customers from C_2 are quite simple to explain. Namely, these customers have activated input neurons that contribute massively to the prediction made. For the customers from C_1 on the other hand, the decision-making is rather based on the entire interaction of the input neurons.

Ideas to Improve the Recommender. A closer look at the most relevant inputs reveals a certain pattern. We have two different types of input neurons: (a) input neurons representing the interaction with a product from P^* and (b) input neurons representing an interaction with a certain product category. In the latter case, an interaction with a category can only take place via an interaction with a product from the associated category. The activation of the categories occurs for each product interaction, regardless of whether or not it is contained in P^*. Now, when looking at the input relevancies for customers from C_1 or C_2, the following is noticeable: Firstly, for customers from C_1, there are no activations of products. The most relevant inputs are therefore categories and the relevancies hardly differ. Secondly, customers from C_2 always have product activations. In these cases, the most relevant input is always a neuron belonging to a product interaction and these relevancies are significantly higher than those of the likewise activated categories. We were thus able to determine that the activation of products as input neurons leads to more unambiguous decision-making. In particular, these represent a better explanatory power as the neural network predicter can identify certain information that significantly influenced the decision made. It would therefore make sense to separate the user information even further and define the products or categories as separate subnetworks. In this way, the decision-making process for user profiles that only contain categories as input neurons could be given a stronger explanatory power.

Understanding the Customer Base. We also evaluated the interior design shop T^2. Our diversity measures σ_{T^2} and $\zeta^+_{T^2}$ revealed that the trained bias and outliers within the positive input relevancies of the neural subnetwork s_2 were particularly relevant for the decisions made. Thus, it was found that buying interest is based on daily trends rather than past interactions. Unfortunately, we cannot explain this in more detail here due to space constraints.

3.5.4 Preliminary Study of Expert Evaluation

We conducted a preliminary study of the expert evaluation. The aim of this study is to identify how the study needs to be conducted in order to design sufficient explanations for further evaluation for end customers of the corresponding shops. In particular, we want to determine the questions and the response scale to present the explanations we get with the LRP method.

Setting. To evaluate the explanations, we created the following role-play. We present two choices from the three most important interactions for the corresponding purchase decision on the most current day. One choice option corresponds to the products with the three highest weights according to the LRP-method. The other choice option is a randomized selection of three products from the previous interactions. The expert then decides between the two possibilities and selects the one that, in his or her opinion, contributed most to the purchase decision on the current day. The interactions of the last 30 days in total are shown. Specifically, we divided the days as follows: 28 days describe the past interactions, 1 day describes the previous day, and 1 day describes the current day with the purchased product.

Design. In order to examine the differences between visual and textual information, we chose two forms of presentation. In the visual form of presentation, the interactions are presented in the form of images. In the textual form of presentation, the interactions are described in a table. With these two forms, we want to find out whether a visual or a semantic presentation contributes to a better understanding. In doing so, we first evaluate 15 examples with images and then 15 examples with text. An example of the survey can be found in Appendix B. The experts in our preliminary study are employees in the field of data science in the respective stores because they know their products best. This means they can assess the extent the best to which the explanations are sufficient. Additionally, they are asked to answer questions regarding persuasion and satisfaction. How strong is the connection between the desire to buy and previous interactions? How easy was it for you to choose a winner? Are you satisfied with the expertise of your winner? Here, we have used a five Likert scale.

Lessons Learned. We drew the following conclusions from our preliminary study. It turned out that 30 case studies (i.e., 15 visual and 15 textual) are too many. Here, we should rather have limited ourselves to five case studies per presentation (i.e., ten in total). Another issue is the scale size. A five Likert scale was too powerful for our study. A three Likert scale would have been sufficient, without loss of information. In addition, it would have been preferable to have many participants to represent the broad mass, rather than just a few experts. Last but not least, it would have been ideal to include the users of the online stores in the research design right at the beginning and have them answer the survey. However, this would mean that the online shops would need to put the survey online in the first place. This will be addressed in future work.

3.6 Conclusion

In this chapter, we have presented TransPer, an explanatory framework for neural networks used in online shopping.

We used the LRP method to define three explanation measures, namely the individuality measure, used to determine those parts of the input that are particularly relevant for the decision; the certainty measure, which measures how certain the system is about its prediction; and the diversity measure, which measures whether there are clear top predictions. These measures can be defined on complex neural networks which process heterogeneous input data.

We have demonstrated the usefulness of our metrics in three explanation use cases. First, we explained fluctuations in the prediction qualities. Understanding the prediction quality facilitates transparency and trust. Second, TransPer explanations can help find ideas on how to improve the neural network which enhances persuasiveness. Third, our explanations can help online shops better understand their customer base which can help satisfy users. These explanations also play an important role in fulfilling legal requirements such as the ones mandated by GDPR. Furthermore, we started an expert evaluation to examine the explanations in terms of their comprehensibility for humans. We wanted to examine the aspect of visual versus textual explanations.

Reconsidering the research questions, we found out that based on the LRP approach, the weights and also the bias is relevant to understand the explanation quality which answers Research Question 1.1. Regarding Research Question 1.2, the explanation quality measures reveal how important specific relevancies are for different customer bases. The leave-one-out method can reveal important features which can be used to improve a neural network. This answers Research Question 1.3.

Outlook. With the preliminary study, we were already able to determine the setting and design in the course of the work. For the future, we would put the evaluation online and let the respective users of the online shops evaluate it directly. These customers are the only ones who can say whether or not the explanation of their personalized recommendation is sufficient. However, we encountered several problems here. This would entail a much greater evaluation, since an explanation is perceived differently depending on the group of people. Since we started with a preliminary study of expert evaluation which are in our case data scientists, a group of non-experts could wish for other explanations. For this reason, it is important that online stores include the evaluation from the very beginning. This way, users are involved from the very start and improvements can be adapted gradually. Nevertheless, this would require the consent of the online shops to integrate the evaluation and run it for a while. Additionally, we need the customers to participate in the evaluation for a certain period of time and repeatedly.

Last but not least, we would integrate our measures into a user-friendly interface for the online shops. Then, they could use it to automatically improve their network if they determine the features to improve. For example, if they find with the individuality measure that the neural network is biased, then they could balance the training data.

4 ObAlEx

The effectiveness of Convolutional Neural Networks (CNNs) in classifying image data has been thoroughly demonstrated. In order to explain the classification to humans, methods for visualizing classification evidence have been developed in recent years. These explanations reveal that sometimes images are classified correctly, but for the wrong reasons, i.e., based on incidental evidence. Of course, images should be classified correctly for the right reasons, i.e., based on the actual evidence.

To this end, we propose a new *explanation quality metric* to measure *ob*ject *al*igned *ex*planation in image classification which we refer to as the ObAlEx metric. Using object detection approaches, explanation approaches, and ObAlEx, we quantify the focus of CNNs on the actual evidence. Moreover, we show that additional training of the CNNs can improve the focus of CNNs without decreasing their accuracy. In particular, we aim to answer the following research questions with our approach.

Research Question 2. Hypothesis: In image classification, object-aligned explanations can sufficiently map the desired explanation of humans and guarantee an intuitive explanation.

2.1 How can right for the right reasons be measured?

2.2 Does it satisfy the concept of classifying right for the right reasons?

2.3 Can it be included in the training process?

This chapter extends work initiated as part of Adrian Oberföll's bachelor's thesis [Obe] and is based on joint work with Adrian Oberföll and Michael Färber [NOF21].

4.1 Introduction

Convolutional Neural Networks (CNNs) have been demonstrated to be very effective in image classification tasks, achieving high accuracy. However, methods to explain classifications performed by CNNs have shown that sometimes image data has been classified for incidental evidence, undermining the trust between humans and machines [RSG16]. Previous attempts to fix this problem have included a human-in-the-loop approach [Sch+20], a pre-processing step for removing features of the input that are deemed irrelevant for the classification task at hand (such as images' backgrounds) [JLC18], or the introduction of a new loss function that incorporates an explanation approach during training [RHD17]. Although the latter work constrains the explanation of the model in the loss function penalizing the input gradients, it uses explanations only based on input gradients which is not ideal for all use cases, especially in image classification, where individual pixels are difficult to interpret. Overall, we believe that there is a lack of a metric that quantifies if an intuitive explanation can be gained.

In this chapter, we propose an *ob*ject *al*igned *ex*planation quality metric, called ObAlEx. ObAlEx quantifies to which degree the object mask of an image is consistent with the obtained evidence of explanation methods and thus, imitates human behavior to classify images according to the objects contained. The proposed metric is independent of the used explanation method (e.g., Deconvnet [ZF14], LIME [RSG16], or Grad-cam [Sel+16]) and object detection method and can therefore be applied together with arbitrary explanation and object detection methods. Our approach to identifying the focus on the relevant input regions requires neither human interaction nor pre-processing. Based on extensive experiments, we demonstrate the effectiveness of the proposed metric while training CNNs, ensuring both high accuracy and a focus on the relevant input regions.

Our main contributions are as follows:

1. We propose an object-aligned explanation metric, ObAlEx, to quantify explanations of image classification models intuitively. Our metric applies to different explanation methods and neither requires human interaction nor interference in the model's architecture.

2. In extensive experiments, we show that our metric can be used for making CNN models for image classification more intuitive while keeping the accuracy. We provide the source code online at https://github.com/annugyen/ObAlEx.

In the following section, we outline our metric. We then present our extensive experiments. Finally, we close with some concluding remarks.

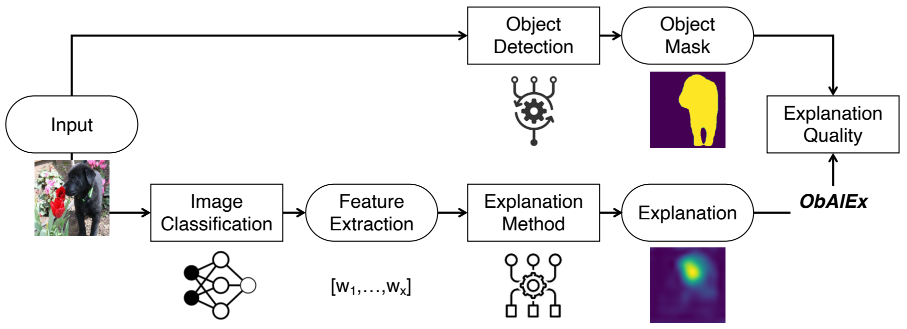


Figure 4.1: The pipeline of our metric.

4.2 OBALEX Metric

The metric OBALEX is designed as a relative metric that depends on the explanation method and the classifier used. Based on the change of the explanation quality during training, it can be evaluated if a certain training strategy leads to an improvement or deterioration of the model's intuitive explanation. By explanation quality, we define the degree of alignment between the object to be classified and the explanation of the classification model.

The pipeline to calculate OBALEX is outlined in Fig. 4.1. Given an input image on which an object should be detected, we first apply an object detection method (e.g., Mask R-CNN) to obtain the image regions of the object itself (i.e., object mask). We define regions of the explanation that lie outside of the object mask as indicative of classification for the wrong reasons, and conversely, that regions of the explanation that lie inside of the object mask as indicative of classification for the right reasons. The mask of objects on images can be obtained with a high accuracy nowadays (see Sec. 4.3).

Simultaneously, an image classifier (e.g., pre-trained VGG16) is applied to obtain labels of recognized objects (e.g., "dog"). An explanation method (e.g., Grad-Cam) then outputs the image regions which are most influential given the extracted features from the CNN and the input image.

Both the object mask and the explanation output is then used to compute the metric OBALEX and thus, to improve the explanation quality. Since existing explanation methods support different highlighting levels, our score is constructed in such a way that the score is the higher the more the highlighted explanation aligns with the object mask. In the following, we describe the computation of the explanation quality formally.

Given a data set D with correctly classified images and an image $d \in D$ with pixels p^d_{ij}, width w^d, and height h^d, let A^d denote the matrix whose values a^d_{ij} equals the activation of the pixels of the object mask, where $i \in \{1, \ldots, h^d\}$, $j \in \{1, \ldots, w^d\}$, $h^d, w^d \in \mathbb{N}$. We regard A^d as a fuzzy set, i.e. whose values have degrees of membership depicted as a^d_{ij}. We define $a^d_{ij} \in \mathbb{R}$ with $0 \leq a^d_{ij} \leq 1$. In our experiments, we set $a^d_{ij} = 1$ if the pixel p^d_{ij} of the input image belongs to the object mask and $a^d_{ij} = 0$, otherwise. Similarly, let B^d be the matrix whose values b^d_{ij} equals the activation of the pixels of the explanation. We additionally normalize the values b^d_{ij} between zero and one, i.e. $0 \leq b^d_{ij} \leq 1$ where $b^d_{ij} = 1$ if the pixel p^d_{ij} of the input image belongs to the highest activation and $b^d_{ij} = 0$ otherwise. Our metric ObAlEx is, then, defined as follows:

$$\text{ObAlEx}(A^d, B^d) = \frac{\sum_{i,j} a^d_{ij} b^d_{ij}}{\sum_{i,j} b^d_{ij}} \in [0, 1] \tag{4.1}$$

To get the explanation quality of an image classifier, ObAlEx can be applied on all images in a data set D. We then calculate the average of all values of the explanation quality of each picture for an image collection. In doing so, we weight all images equally. The explanation quality of the classifier is defined as

$$\text{AvgObAlEx}(D) = \frac{1}{n} \sum_{d=1}^{n} \text{ObAlEx}(A^d, B^d) \in [0, 1], \tag{4.2}$$

where $n \in \mathbb{N}$ is the number of images in data set D. AvgObAlEx only considers the scores of images classified correctly by the model, otherwise the metric would get skewed. Therefore, images which are classified wrong are excluded.

4.3 Evaluation

4.3.1 Evaluation Setting

To evaluate ObAlEx, we apply pre-trained CNN models. We focus on three state-of-the-art image classification models: *VGG16* [SZ15], *ResNet50* [HZRS16], and *MobileNet* [How+17]. The models are pre-trained on the ILSVRC2012 data set [Rus+15] which is also known as ImageNet. We adapt each model's upper output dense layers to the specific data set (i.e., number of categories in the used image classification data sets *Dogs vs. Cats* and *Caltech 101*, respectively). To show the universal applicability of ObAlEx, we use different well-known explanation methods such as Deconvnet [ZF14], LIME [RSG16], Grad-Cam [Sel+16], and Grad-Cam++ [CSHB18]. In our experiments, the AvgObAlEx settled around a fixed value after 50 images. For that reason and due to high computing power costs in the case of LIME, we calculate the AvgObAlEx for 50 images per epoch in the following experiments. Our experiments are executed on a server with 12 GB of GPU RAM. We use TensorFlow and the Keras deep learning library for implementation. We use the following data sets in our evaluation:

- **Dogs vs. Cats** data set[2] contains 3,000 dog and cat images, 1,500 per class. We use Mask R-CNN [HGDG20] to create the object masks. The quality of the object masks is important for the validity of the proposed metric ObAlEx. Therefore, we manually evaluated the computed object masks for 200 randomly chosen images regarding the overlap of the whole object. The (top-1) accuracy was 91%. Thus, we argue that the pre-trained Mask R-CNN performs well for our purpose.

  Given the data set size, we used 70% of the images for training and 30% for testing. We first adjust the output layer of all CNN models to the two categories (dog and cat) and train them for 10 epochs on the Dogs vs. Cats data set (where all layers except the output layer are frozen). After that, we freeze different combinations of layers for further training. In the original papers of the above-mentioned models, the convolutional layers are divided into five blocks. For simplification and comparability, we use this convention for our strategies. We also summarize the last dense layers to one block. Thus, we always set whole blocks of layers to either be trainable or non-trainable. We train every strategy for another 10 epochs. We investigate the following strategies: (a) train the last dense layers which we denote as dense blocks, (b) train the last two convolutional blocks (i.e. the fourth and fifth), (c) train the first three convolutional blocks, and (d) train all layers, i.e. all convolutional and dense blocks.

- **Caltech 101** data set [FFP07] has 101 object categories. We create a uniform distributed data set by drawing random sampling from the categories resulting in a total of 6,060 images with 60 images per class. We use a test split of 0.25. This data set is provided with hand-labeled object masks for all images. Thus, we use those labeled object masks. We perform another experiment inspired by [RHD17, Sch+20]. To actively force the model to be more intuitive and thus, to provide a more interpretable explanation, we followed a naïve approach by using artificial images. We edit the images in a way that they contain the object to classify and masked out the background with random pixels. This should force the model to focus more on the object and increase the explanation quality.

[2] https://www.kaggle.com/c/dogs-vs-cats, last accessed: 2020-10-28

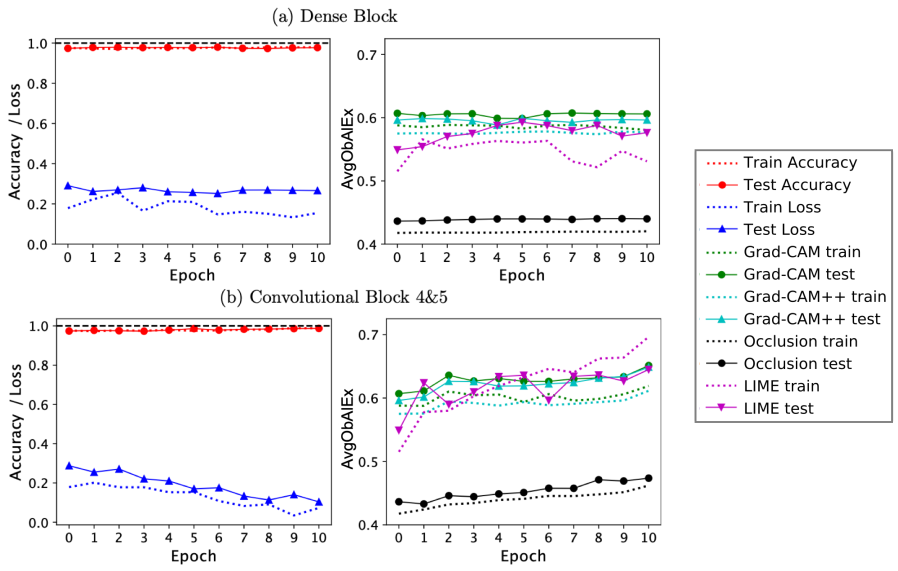


Figure 4.2: VGG16 Results. Transfer learning strategies with VGG16 with explanation methods Deconvnet (Occlusion), LIME and Grad-Cam/Grad-Cam++.

4.3.2 Evaluation Results

Dogs vs. Cats. Fig. 4.2 shows the results for VGG16 with training strategies (a) and (b). We can see that the performance of the model measured with accuracy did not change within 10 epochs (see Fig. 4.2 (a)/(b) left graph). However, we observed a change in AvgObAlEx (see Fig. 4.2 (a)/(b) right graph). The explanation quality after 10 epochs computed with any explanation method for strategy (b) is significantly higher than the explanation quality for strategy (a). This fits to the common knowledge that complex structures in the input images are learned in the later convolutional blocks and are, therefore, more decisive for the classification. Moreover, Fig. 4.2 (b) shows with an increasing number of epochs a decrease in the loss, while the AvgObAlEx increases simultaneously. This indicates the effectiveness of the model for the right predictions based on the right reasons. The results of strategy (d) and (b) and the results of strategy (c) and (a) are similar to each other respectively, which emphasizes the common knowledge. Without using the proposed metric ObAlEx this improvement would not be evident since the accuracy of all models stays the same during training.

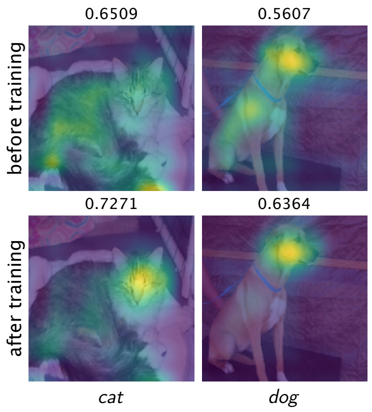


Figure 4.3: Examples from Dogs vs. Cats with quality scores shown above.

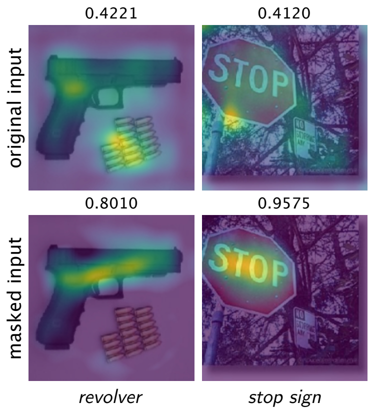


Figure 4.4: Examples from Caltech 101 with quality scores shown above.

In Fig. 4.3, we provide an example of the explanation visualized with Grad-Cam with strategy (b) on VGG16. We can see that the explanation quality increases after training and that the visualized explanation has a stronger focus on the object. With only 10 epochs of additional training, we were able to improve the model in a way that it utilizes more important features such as the face of the animal. Without ObAlEx, it would be obvious to not train the model any further due to the non-changing accuracy. We observed similar results on the experiments with ResNet50 and MobileNet, and also on the Caltech 101 data set.

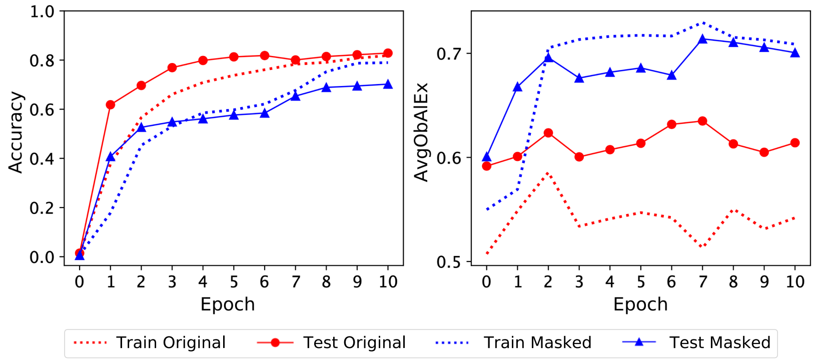


Figure 4.5: Training on Caltech 101.

Caltech 101. Fig. 4.5 shows the results for 10 epochs of training VGG16 on Caltech 101 with the original and masked images as input. As we can observe in the left graph, training with the original images results in higher accuracy than training with the masked images. However, the AvgObAlEx (computed with Grad-Cam as explainer, see graph on the right) of the model trained with masked input images is significantly higher than the AvgObAlEx of the model trained with the original input images. This indicates that more background information was used in the classification. Thus, evaluating image classifiers beyond accuracy can be valuable to real-world cases where specific background information is unavailable.

Fig. 4.4 shows an example image with Grad-Cam on VGG16. Despite high accuracy, we can see that the explanation for the image with masked out background (image at the bottom) is more intuitive and more focused on the actual object than the original input image.

4.4 Conclusion

In this chapter, we focused on evaluating CNN image classifiers with different explanation approaches. We introduced a novel explanation quality score metric to support the training process besides accuracy and loss function which facilitates scrutability. We have shown in our experiments that our metric ObAlEx can be used to effectively indicate cases where a model makes its predictions based on wrong reasons. Overall, ObAlEx facilitates more generalized models which can increase the user's trust in the model by object-aligned explanations.

Back to the research questions, the ObAlEx metric measures the alignment of the explanations with the actual object and thus can be used to measure the right features for the right reasons. This answers Research Question 2.1. By considering different state-of-the-art explanation methods in our approach, the ObAlEx metric satisfies the concept which answers Research Question 2.2. Considering Research Question 2.3, the ObAlEx metric can be calculated during the training process to see the improvement of the explanations in addition to the accuracy and loss function.

Outlook. As future work, we want to integrate ObAlEx in a loss function. As we used different explanation methods, we have to commit to one to define a score depending on the used explanation approach. Furthermore, we only applied ObAlEx on image data. An extension on other data types would be obvious.

5 FilTag

Convolutional neural networks (CNNs) have achieved astonishing performance on various image classification tasks, but it is difficult for humans to understand how a classification comes about. Recent literature proposes methods to explain the classification process to humans. These focus mostly on visualizing feature maps and filter weights, which are not very intuitive for non-experts in analyzing a CNN classification.

In this chapter, we propose FilTag, an approach to effectively explain CNNs even to non-experts. The idea is that when images of a class frequently activate a convolutional filter, then that filter is tagged with that class. These tags explain a reference of a class-specific feature detected by the filter. Based on the tagging, individual image classifications can then be intuitively explained in terms of the tags of the filters that the input image activates. Finally, we show that the tags are helpful in analyzing classification errors caused by noisy input images and that the tags can be further processed by machines. In particular, we aim to answer the following research questions with our approach.

Research Question 3. Hypothesis: Filters in CNNs encode semantic information of the input images.

3.1 How can the encoded semantic information of the filters be decoded?

3.2 How precise are the tagged filters in predicting and understanding the output of the CNN (compared to visual methods)?

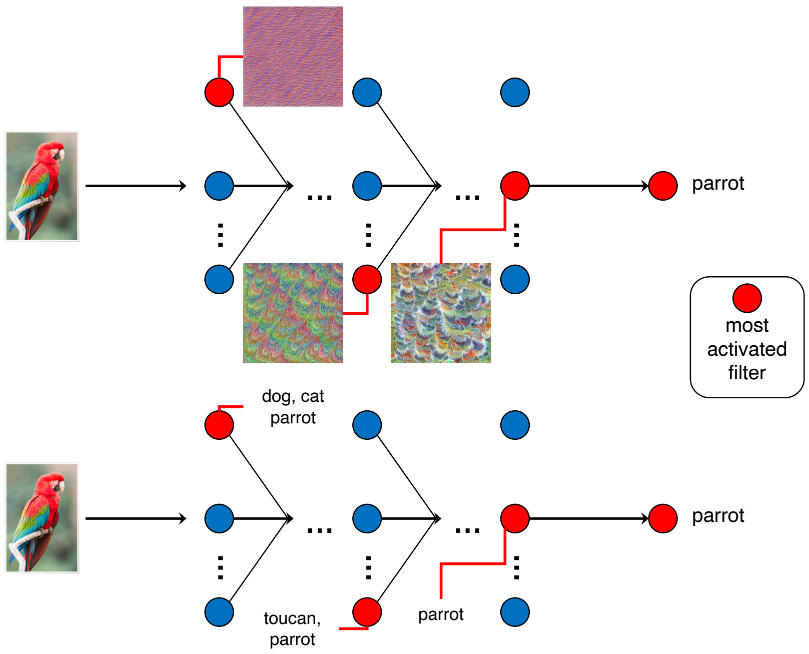


Figure 5.1: Explanations of Convolutional Filters. The upper part shows a visual explanation. The lower part contains an example of our tagging approach FilTag.

3.3 What benefit does link tagged filters to knowledge graphs offer?

This chapter extends work initiated as part of Daniel Hagenmayer's master's thesis [Hag] and is based on joint work with Daniel Hagenmayer, Tobias Weller and Michael Färber [NHWF21].

5.1 Introduction

Deep convolutional neural networks (CNNs) are the state-of-the-art machine learning technique for image classification [SZ15, Sze+16]. In contrast to traditional feed-forward neural networks, CNNs have layers that perform a convolutional step (see Figure 5.2 for the relations in a convolution). Filters are used in a convolutional step which outputs a feature map in which activated neurons highlight certain patterns of the input image. Although CNNs achieve high accuracy on many classification tasks, these models do not explain (i.e., decisive information) the classifications. Thus, researchers recently focused on methods to explain how CNNs classify images.

Related Work. Some of the earliest works on explaining CNNs focus on visualizing the activations of individual neurons [MSM18, OMS17]. However, these methods cannot explain more complex relationships between multiple neurons, as no human-understandable explanation is used. Olah et al. [Ola+18] defined a semantic dictionary by pairing every neuron activation with its abstract visualization using a channel attribution, determining how much each channel contributes to the classification result. This may explain the role of a channel in the classification of an individual image, but it does not explain the role of that channel across all possible input images. Hohman et al. [HPRC20] try to overcome this problem by aggregating particularly important neurons and identifying relations between them. Other approaches focus on filters, the discerning feature of CNNs. For example, Zeiler and Fergus [ZF14] visualize the filter weights to illustrate the patterns these filters detect. However, these visualizations are based on the inputs of the layers to which the respective filter belongs. Thus, only the filter patterns of the first layer can be directly associated with patterns on the input image of the network. To overcome this, the method Net2Vec [FV18] quantifies how concepts are encoded by filters by examining filter embeddings. Alternatively, Network Dissection [Bau+17] uses human-labeled visual concepts to bring semantics to the convolutional layers. However, visualizations and embedding filters only explain the outcome of a model implicitly, whereas we assign explicit tags to filters that can be understood by non-experts. Most visualizations used for explaining CNNs are similar to the example in Figure 5.1, which visualizes the most activated convolutional filters. Such visualizations are difficult to understand on their own. Adding an explicit explanation such as a semantic tag (e.g. dog, parrot, cat, or toucan) as shown in the bottom example would dramatically improve the explanation, including for non-experts.

Contribution. Our contribution is threefold. First, we introduce FilTag, an automatic approach to explain the role of each convolutional filter of a CNN to non-expert humans. We use the fact that each filter is dedicated to specific sets of classes [ZF14, GDDM14, SVZ14, SDBR15]. Indeed, the idea of FilTag is to quantify how much a filter is dedicated to a class and then tag each convolutional filter with a set of, particularly important classes. The lower part of Figure 5.1 shows an example of what a CNN tagged in this way could look like. In that example, the rightmost filter highlighted in red plays a role in classifying parrots, whereas the filter in the middle only plays a role in classifying birds in general, as both, toucans and parrots are birds. This filter extract features that are specific to these classes (e.g. wings, feathers, etc.).

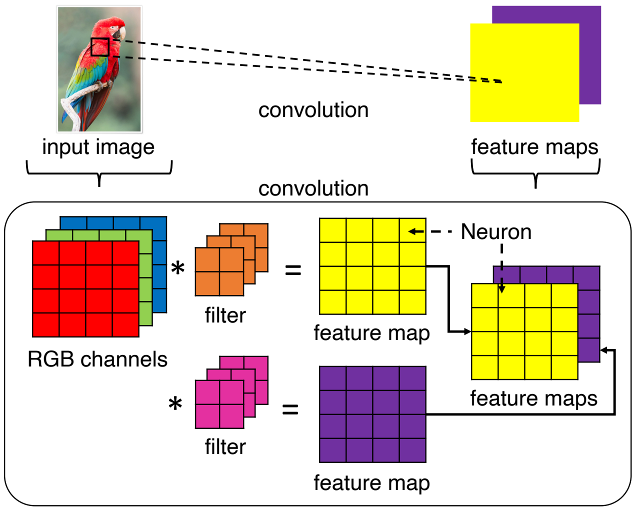


Figure 5.2: Terminology and basic relations of a filter in a convolution.

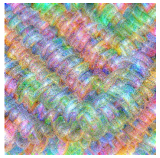
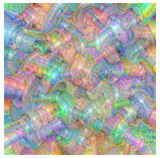
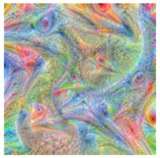
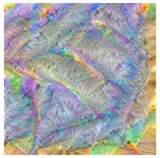
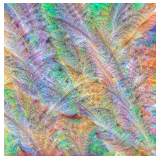
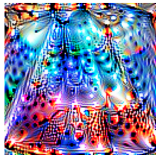
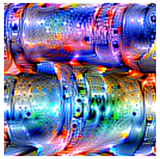
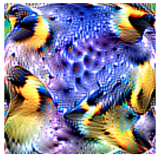
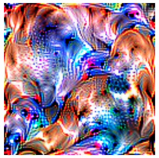
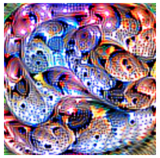
Second, our approach can also be used to explain the classification of an individual image. In the example in Figure 5.1, the classification of the input image as a parrot would be explained by the union of the tags of the activated filters, which are all animals, particularly tagged with a parrot. Third, FilTag is suitable to analyze classification errors. We analyze our approach with thorough experimentation using ImageNet as a data set and using multiple CNNs, including VGG16, VGG19, and InceptionNet with pre-trained models. We focus on the experiment on VGG16. The source code is available online[3].

5.2 Approach

Our approach consists of three components. In Section 5.2.1, we explain the role of each filter in a CNN (independent from concrete input images) using our concept of filter tags. Then, in Section 5.2.2, we explain how a particular input image can be explained, namely in terms of the filters that it activates. Finally, we show how to use filter tags to analyze classification errors in Section 5.2.3.

[3] https://bit.ly/3hjeMR4

Table 5.1: Comparison of filter explanations from the last convolutional layer from VGG16 [SZ15].

Approach	Filter 95	Filter 150	Filter 288	Filter 437	Filter 462
Erhan, Bengio, Courville, and Vincent 2009					
Olah, Mordvintsev, and Schubert 2017					
FILTAG (k=1)	table lamp, yurt, lampshade	cannon	goldfinch, toucan, european fire salamander	rhodesian ridgeback, bloodhound, redbone, *other dogs*	rock python, boa constrictor
pictures corresponding to the tags					

5.2.1 Explanations of Filters

Our explanation of filters works in two steps. In the first step, we quantify how much each filter is activated by images of each class. In the second step, we use this information to tag the filters.

Quantifying Filter Activations. Feature maps with high activations can be used as an indication of the importance of the preceding filter for the input image [HPRC20, ZF14]. Traditional explanation approaches focus on one image and therefore use the most activated feature map while our approach focuses on a set of images of the same class.
Given a pre-trained CNN with a set of convolutional layers M with its respective set of filters $I_{(\cdot)}$ and a labeled data set D with labels $c \in C$ from a set of labels C, let $d \in D$ be an input image and $m \in M$ a convolutional layer. First, we collect the activations in the feature map to get the importance of the filters regarding an input image, i.e. the output in the feature map for a given filter (see terminology in Figure 5.2). Second, we scale these activations per layer between $[0, 1]$. In scaling the activations, we ensure that no image is overrepresented with overall high activation values. We scale the activations per layer because each layer has its specific pattern compositionality of filters. For example, the first convolutional layers detect simple patterns such as lines and edges whereas the layers, in the end, detect compositional structures which match better to human-understandable objects [ZF14]. Let $a(m, i, d, j)$ be such a scaled activation in the jth element in the feature map calculated from image d and filter $i \in I_m$ in convolutional layer m. In order to get a total activation value per feature map, we define $\bar{a}(m, i, d)$ as the arithmetic mean of the scaled activations in a feature map:

$$\bar{a}(m, i, d) = \frac{1}{n} \sum_{j}^{n} a(m, i, d, j), \quad 0 \leq \bar{a}(m, i, d) \leq 1, \tag{5.1}$$

where n is the number of activations in the feature map. We do this for all filters $i \in I_m$ and repeat these steps for all layers $m \in M$.
Next, we use the labels as the desired explanation. Let d_c be an input image with label c. We define $z_c(m, i)$ as arithmetic mean of $\bar{a}(m, i, d_c)$ over one class c:

$$z_c(m, i) = \frac{1}{|D_c|} \sum_{d_c}^{|D_c|} \bar{a}(m, i, d_c), \quad 0 \leq z_c(m, i) \leq 1, \tag{5.2}$$

where $|D_c|$ is the number of images in class c. This way, $z_c(m, i)$ is the averaged value of all activations of the images in one class respective its filter i in layer m. Thus, we can rank the classes according to the highest averaged activation of the filter per layer which will be the decisive criterion for the labeling. We, therefore, compare the received values for each feature map. We repeat these steps for all images in D per label class.

Filter Tagging. We tag the filters according to their corresponding values received in Equation 5.2 with the label of the input image class. We are interested in the feature maps with high activations of a certain class because they indicate important features associated with that class [HPRC20]. We define two methods to select those feature maps per class and per layer (because of the mentioned complexity in different layers):

i) k-best-method: choose the k feature maps with highest activation values.

ii) q-quantile-method: choose the q-quantile of feature maps with highest activation values.

These tags serve as an explanation of what the filter does. For example, in Figure 5.1, the leftmost activated filter has the three tags *dog*, *parrot* and *cat*, which suggests that this filter plays a role in recognizing animals.

5.2.2 Explanations of Individual Classifications

While previous visual methods for explaining filters are difficult for humans to understand, a textual assignment can lead to unambiguous explanations (as later seen in our experiments in Figure 5.1). To get an explanation given input, we assume that the tags have a better information value with the classification of the CNN if the tags match with the classification output. Therefore, we want to measure the hit of the prediction with the tags in the most activated filters. To do this, we determine the most frequently occurring labels for each image of a class according to the previously mentioned method using the metric Hits@n. Hits@n measures how many positive label tags are ranked in the top-n positions. For example, in Figure 5.1, the classification of the input image as a parrot is explained by its high activation of filters tagged with *parrot*.

5.2.3 Analysis of Classification Errors

FilTag can be used for error analysis using Hits@n. Taking misclassified input images, Hits@n indicates if the most relevant filters were activated. If Hits@n is high, we can assume that there are similar features of the misclassified class and original image. Analyzing the tags, we may find correlations in their semantics. Furthermore, linking the tags and filters to knowledge graphs such as ConceptNet [SCH17] or FAIRnets [NWFS20] can bring more insights. ConceptNet is a semantic network with meanings of words and FAIRnets is a neural network graph with meta-information about the architecture. For example, in Figure 5.1, if we input a picture of a car but the most activated filters have tags of animals, we can conclude that the wrong filters were activated.

5.3 Experiment

5.3.1 Experimental Setup

Data Set. Following related work, we use ImageNet [Rus+15] data set from ILSVRC 2014 to conduct experiments on the introduced approach. This data set contains over one million images and 1,000 possible class labels including animals, plants, and persons. Each class contains approximately 1,200 images. We use a holdout split, using 80% of the images to tag the filters while ensuring that there were at least 500 images from each class in the set, and the remaining 20% to test the explanations.

Baseline. We compare our approach with two state-of-the-art visualization methods in explaining neural networks. The selection of the methods was based on their focus on feature visualization. One of the methods used provided the fundamental basis of visualization of features and uses minimal regularization [EBCV09], the other method uses optimization objectives [OMS17].

Implementation. We implemented our method in Python3 and used TensorFlow as a deep learning library. The experiments were performed on a server with Intel(R) Xeon(R) Gold 6142 CPU@2.60 GHz, 16 physical cores, 188GB RAM, and GeForce GTX 1080 Ti. We used pre-trained neural network models from Keras Applications. The filters of a VGG16 were explained in the experiments using the introduced method. VGG16 was used as CNN as it is frequently used in various computer vision applications. The evaluation for VGG19 and InceptionNet gave similar results and can be executed with the given code.

5.3.2 Analysis of the Explanations

In this analysis, we want to study the explanations of the filters using k-best-method, with $k = 1$, to provide a better comparison with the state-of-the-art methods since they frequently visualize the most activated feature map. Figure 5.1 shows exemplarily the visual explanations of the baseline methods and the tags of our approach FilTag.

As shown, the visual explanations of the baseline methods [EBCV09, OMS17] do not provide satisfactory comprehension. At first sight, there is not much to understand. Considering our tags, one can imagine what the visualizations display. We additionally include pictures corresponding to our tags, to show the information value compared to only visualizations of the filters. Filter 95 seems to recognize a lampshade especially a trapezoidal shape. Filter 150 is only tagged with *cannon*, i.e. the filter is specific for this class. Filter 288 detects the head of a goldfinch especially with consideration of the yellow and black pattern. Filter 437 and Filter 462 recognize the ears of brown dogs and the body of snakes, respectively. This information would be hard to retrieve without the tags. Even without considering the visualizations, one has a good impression of what a filter detects. For example, it is quite impressive that Filter 288 detects this black yellow pattern which we can follow from the tags *goldfinch, toucan*, and *european fire salamander.* As well, Filter 95 detects the trapezoid in *table lamp, yurt*, and *lampshade.*

In addition to comparing our method to the state-of-the-art methods in CNN explanations, we linked the tags to concepts from ConceptNet [SCH17] to achieve a coarsening of common tags. ConceptNet is a semantic network with the meanings of words. This comparison revealed that many tags have both visual and semantic commonalities (e.g., see Filter 437 in Figure 5.1, rhodesian ridgeback, bloodhound, and redbone are all of the type dogs). Following this evaluation process, we manually reviewed 100 filters in the context of common visual and semantic commonalities. Here we found 88% conformance with common tags in the filters.

5.3.3 Impact of Hyperparameters

In this evaluation, we show the impact of the hyperparameters k and q on interpretability and expressiveness. In Figure 5.3, we compute Hits@n with the test set from ImageNet depending on k and q. We can see that Hits@n increases for increasing k, q and n. For $q = 25\%$ and $n = 50$, we even get a hit rate of 80% over all 1,000 object classes. This result shows that FilTag can be taken as a significant explanation for the classification. For example, we have observed that the class *shoji* gets the highest hit rate of 98.47% followed by the classes *slot, odometer, entertainment center*, and *bookshop* with also around 98%. The classes with worst hit rates are *spatula* (51.19%), *schipperke* (50.97%), *reel* (49.8%), *bucket* (46.83%), and *hatchet* (36.75%).

The corresponding likelihoods of the best classes are shoji (81.22%), slot (92.30%), odometer (91.73%), entertainment center (82.89%), and bookshop (66.41%). Likewise, the accuracies of the worst classes are spatula (30.15), schipperke (71.81%), reel (57.25%), bucket (48.80%), and hatchet (50.60%). These results fit to the top-1 accuracy of VGG16 with $74,4\%$ for all classes.[4] However, for larger values of q we observed that the interpretability decreases because the number of tags increases for each filter.

[4] https://paperswithcode.com/sota/image-classification-on-imagenet, last accessed: 2021-03-02.

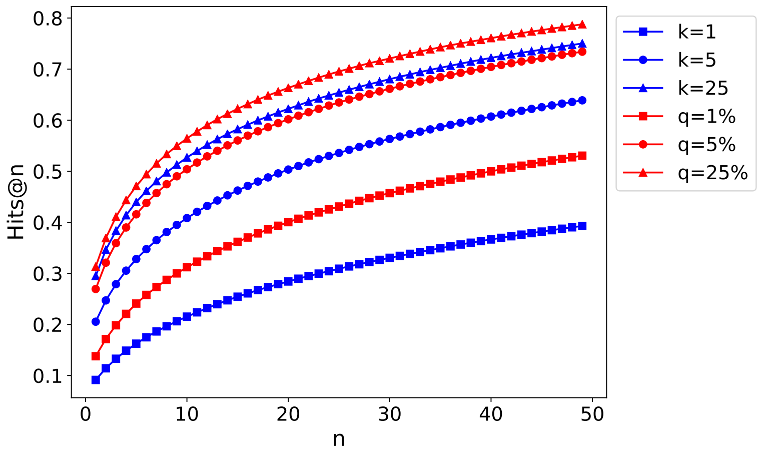


Figure 5.3: Hits@n with different k and q on ImageNet

This makes it harder to find similarities between the classes. Thus, there is a trade-off between expressiveness for the classification and interpretability for the filters.

5.3.4 Using the Explanations

FilTag can be used for error analysis using Hits@n. Taking misclassified input images, Hits@n indicates if the most relevant filters were activated. If Hits@n is high, we can assume that there are similar features of the misclassified class and original image. Analyzing the tags, we may find correlations in their semantics.

Figure 5.4 shows an image of the class *mortarboard* in ImageNet. Using VGG16, the class *academic gown* is predicted with confidence of 83.8%, while the actual class *mortarboard* is predicted with a confidence of only 16.2%. Considering the image, we notice that both objects are part of this image, making this result reasonable. Reviewing the activated filters, we observe that filters tagged by FilTag with the tag *mortarboard*, as well as with the tag *academic gown*, are usually activated. As a result, we can verify that features are extracted from these two classes and used for prediction. This allows giving non-experts an understanding of the reason for the misclassification, as often features of the other class are extracted from this image. Likewise, we can use the information to increase the number of images in which the mortarboard is the actual class but not in the main focus of the image, to continue training the network to make the predictions more accurate.

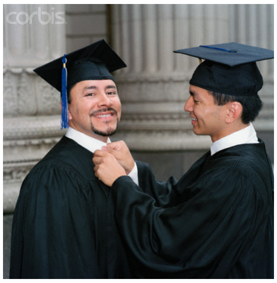


Figure 5.4: Example image of class mortarboard

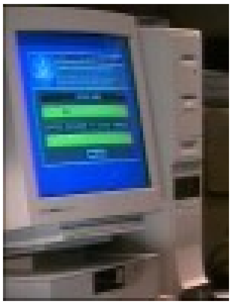


Figure 5.5: Example image of class desktop computer

Figure 5.5 shows an image from the class *computer*. This image is classified by VGG16 as *cash machine* with a probability of 99%. Looking at the tagged filters, filters of the tags *cash machine* are mostly activated, followed by *screen*, *CD player*, and *file*. Considering Figure 5.5 and having knowledge about the other images of the class *computer* in ImageNet, the reason this image is not assigned to this class becomes clear. Generally, frontal images of a computer were used for the *computer* class for learning. However, this image does not correspond to the same distribution. Thus, it is very difficult for the neural network to assign it correctly. Moreover, it is a very old computer, whereas the other images in ImageNet generally represent rather modern computers. To classify this image correctly, further images showing old computers from the side have to be included to change the distribution and train the VGG16 to classify this image correctly.

5.4 Conclusion

We have introduced FilTag, an approach to provide human-understandable explanations of convolutional filters and individual image classifications. These tags can be used to query and identify specific filters that are relevant for feature detection. In contrast to state-of-the-art explanations, our approach allows for explicit, non-visual explanations which are more understandable for non-experts. Moreover, we have demonstrated that FilTag can be used to understand and analyze classification errors. This improves scrutability and thus human understanding of CNNs which contributes to making artificial intelligence more trustworthy.

Coming back to the research questions, we could show that by tagging the filters of convolutional layers, the encoded semantic information of the filters can be decoded in human-understandable language. This answers Research Question 3.1. Regarding Research Question 3.2, we found out that words can be more comprehensible compared to visual methods. We could show with our evaluation that the tagged filters are quite accurate in predicting and understanding the output of the CNN. Considering Research Question 3.3, linking the tagged filters to other knowledge graphs allows for expanding the information of the labeled filters.

Outlook. We plan to extend our evaluation for general classification tasks using CNNs, e.g. not only for images but also texts.

The approach described in this chapter uses ImageNet class labels as tags. These class labels name concrete objects such as “parrot”, “car”, or “table lamp”. Some of these objects may share commonalities. For instance, parrots, goldfinches and toucans are all different kinds of birds. It would be interesting to link the ImageNet class labels together with a knowledge graph that describes the relationships between these objects, such as ConceptNet. In such a knowledge graph, the vertices “parrot”, “goldfinch” and “toucan” could all be connected to the vertex “bird” by an “is a kind of”-edge. Using this information, can we use the filters tagged with parrot, goldfinch and toucan to determine filters that recognize birds?

A similar but slightly different angle is that parrots and toucans have eyes, a beak and a tail. Cats do not have a beak, but they also have eyes and a tail. Again, this could be represented in a knowledge graph by a “has a”-edge linking these animals with the vertices representing eyes and tails. Can we use these relationships to determine filters that recognize eyes or tails?

We note that providing either the positive or the negative answer to these questions is very interesting. In the positive case, we obtain a very powerful system that would provide classification explanations such as “This is an image of a parrot because it features eyes, a beak and a tail”. Such a system would be highly persuasive to humans. In the negative case, we obtain evidence that CNNs do not recognize objects in a way that is reminiscent of deductive reasoning but in a fundamentally different way, leading to questions about how neural networks perceive objects.

We have made some preliminary excursions in this direction. They have revealed that the integration of ImageNet's roughly one thousand classes (many of which are highly specific) into ConceptNet is not sufficiently tight to answer the questions above. Expanding ConceptNet appropriately with manual annotations could mitigate this problem.

6 FAIRnets

Research on neural networks has gained significant momentum over the past few years. Because training is a resource-intensive process and training data cannot always be made available to everyone, there has been a trend to reuse pre-trained neural networks. As such, neural networks themselves have become research data. The problem with the information basis on neural networks available online is that they are represented in different formats, repositories, and information bases such as only model architecture. The FAIR guiding principles encourage to provide meta-information to make digital assets more transparent to especially non-experts in computer science such as data scientists.

In this chapter, we propose an approach that summaries the neural networks according to the FAIR guiding principles by proposing an ontology that enables semantic annotations to enhance the information basis and representing a wide range of existing neural network models with this ontology in a knowledge graph. To simplify the search of our knowledge graph, we implemented a search engine for neural networks with over 18,400 neural networks. In particular, we aim to answer the following research questions with our approach.

Research Question 4. Hypothesis: Neural network models which comply with the FAIR Principles support humans in dealing with the increase in volume, complexity, and creation speed of these models.

4.1 Which information should be provided about the neural network models to enable transparency?

4.2 Which information must be provided to apply an existing model to a novel use case?

4.3 What kind of functions should be provided for supporting humans in reusing neural networks?

This chapter is based on joint work with Tobias Weller, Michael Färber, and York Sure-Vetter [NWFS20, NW19].

6.1 Introduction

Researchers of various sciences and data analysts reuse but also re-train neural network models according to their needs. Providing pre-trained neural network models online has the following advantages. First, as a provider, you can benefit from users improving your neural network and circulating your research. Second, as a user of an already trained neural network, you can overcome the cold start problem as well as save on training time and costs. Furthermore, providing trained neural network models gets increasingly important in the light of the research community's efforts to make research results more transparent and explainable (see FAIR principles [Wil+16]). As a result, more and more trained models are provided online at source code repositories such as GitHub. The models provided serve not only to reproduce the results but also to interpret them (e.g., by comparing similar neural network models). Lastly, providing and using pre-trained models gets increasingly important via transfer learning in other domains.

To ensure the high-quality reuse of data sets and infrastructure, the *FAIR Guiding Principles for scientific data management and stewardship* [Wil+16] have been proposed. These guidelines are designed to make digital assets **F**indable, **A**ccessible, **I**nteroperable, and **R**e-usable. They have been widely accepted by several scientific communities nowadays (e.g., [Wis+19]). Making digital assets FAIR is essential to deal with a data-driven world and thus keeping pace with an increasing volume, complexity, and creation speed of data. So far, the FAIR principles have been mainly applied when providing data sets and code [Wis+19, DPGK19], but not machine learning models, such as neural network models. In this chapter, we bring the FAIR principles to neural networks by (1) proposing a novel schema (i.e., ontology) which enables semantic annotations to enhance the information basis (e.g., for search and reasoning purposes), (2) representing a wide range of existing neural network models with this schema in a FAIR way, and (3) providing an online search service which facilitates queries indicating the desired characteristics of the neural network. As we outline in Section 6.4.1, extracting metadata from neural networks automatically is a nontrivial task due to heterogeneous code styles, dynamic coding, and varying versioning. The key idea is that the information contained in these networks should be provided according to the FAIR principles. This comprises several steps which not only consist of having identifiers but providing (meta)data in a machine-readable way to enable researchers and practitioners (e.g., data scientists) easy access to the data. We facilitate this by using semantic web technologies such as OWL and RDF/RDFS.

Overall, we provide the following contributions:

1. We provide an *ontology*, called FAIRNets Ontology, for representing neural networks. It is made available using a persistent URI by w3id and registered at the platform Linked Open Vocabularies (LOV).
   Ontology URI: https://w3id.org/nno/ontology
   LOV: https://lov.linkeddata.es/dataset/lov/vocabs/nno

2. We provide a *knowledge graph*, called FAIRNets, representing over 18,400 publicly available neural networks, following the FAIR principles. FAIRNets is available using a persistent URI by w3id and is uploaded to Zenodo.
   Knowledge Graph URI: https://w3id.org/nno/data
   Zenodo: https://doi.org/10.5281/zenodo.3885249

3. We provide a *search service*, called FAIRNets Search, to query, search and find neural networks in our knowledge graph.
   Search URI: http://km.aifb.kit.edu/services/fairnets/

The chapter is structured as follows. Section 6.2 gives an overview of related work. Section 6.3 describes the structure of FAIRNets Ontology and Section 6.4 describes the knowledge graph FAIRNets. Section 6.5 explains the reason why the neural networks in FAIRNets follow the FAIR principles. Section 6.6 demonstrates use cases possible with FAIRNets Search. Section 6.7 describes the impact of FAIRNets. Lastly, the contributions are summarized.

6.2 Related Work

Information of neural network models. Mitchell et al. [Mit+19] suggest which information about neural networks should be considered as relevant when modeling them. Information such as description, date of the last modification, link to papers, or other resources to further information, as well as the intended purpose of a neural network, are taken into account. Storing such information makes neural networks more transparent. We follow this suggestion by defining a semantic representation that, to the best of our knowledge, does not exist for neural network models so far.

The knowledge extraction from neural networks can point out relevant features or redundancies [BG97]. We extract neural network information to build a knowledge graph to better evaluate the causal relationships between different neural network architectures.

Representing and provisioning neural network models. There exist several standards for the exchange of neural network information on the instance level. The Predictive Model Markup Language (PMML) [Gro] is an XML-based standard for analytic models developed by the Data Mining Group. PMML is currently supported by more than 30 organizations. The Open Neural Network eXchange format (ONNX) [Fou] is a project by Facebook and Microsoft that converts neural networks into different frameworks. These two formats serve as an exchange format for neural networks on the instance level. We are less interested in the exchange of formats, but rather the reusability of the neural networks on a meta-level. Therefore, our FAIRnets Ontology lifts its elements to a semantic level, i.e. to RDF/S, following a methodology for reusing ontologies [PM00] and applying the Linked Data Principles [Ber]. Thus, we incorporate information on the instance and meta-level in the knowledge graph FAIRnets.

Neural network repositories. Many pre-trained neural networks are available online. The well-known Keras framework [Ker] offers ten pre-trained neural networks for reuse. The Berkeley Artificial Intelligence Research Lab has a deep learning framework called Caffe Model Zoo [BAI] which consists of about fifty neural networks. Wolfram Alpha has a repository with neural networks [Alp] which consists of approximately ninety models. These pre-trained neural networks are represented in different formats making it, for instance, difficult to compare or reuse neural networks. Besides, a larger number of neural networks can be found in code repositories such as GitHub. These neural networks are typically coded in one of the major programming frameworks such as Keras, TensorFlow, or PyTorch. Our approach aims to consider such neural networks and make them available as FAIR data.

6.3 FAIRnets Ontology

6.3.1 Creation Process

The FAIRnets Ontology is dedicated to model metadata for neural network models on a schema level. We developed the ontology by using Protégé [Mus15]. To the best of our knowledge, there is no existing vocabulary for the specific description of neural networks. That is why several senior researchers use best practices [GP09] to construct the ontology. We identify researchers, especially beginners, as potential users. The use cases we envision can be found in Section 6.7.

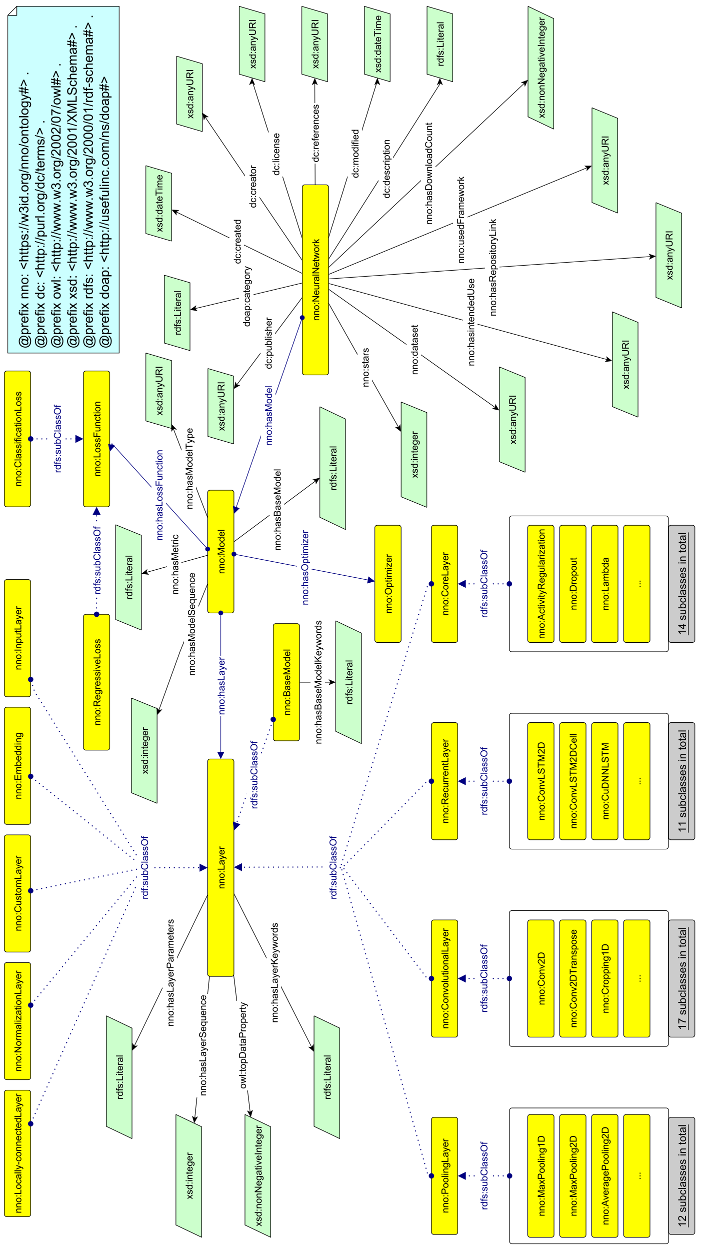


Figure 6.1: Visualization of FAIRnets Ontology.

In addition to the consideration of the Predictive Model Markup Language (PMML) in the development of the ontology (especially in describing the architecture), findings from further work were also considered. In particular, model cards [Mit+19] were taken into account to validate relevant concepts. Model cards encourage transparent model reports concerning machine learning models and are used for outlining the intended use of a model. These cards define minimal information needed to sufficiently describe a machine learning model (in our case, a neural network) that is relevant to the intended application domains. As suggested from model cards, we included model details such as the person developing the model, model date, model type, and licenses.

Characteristics

The structure of the FAIRnets Ontology can be seen in Figure 6.1. Overall, the ontology consists of a total of 516 axioms and uses a total of 77 classes where 70 are sub-classes. It also consists of four object properties, 23 data properties, and 29 individuals.

The ontology enables representing three different aspects of information. (1) Neural network-related general metadata and (2) neural network-dependent features can be modeled, such as the type of layer, loss function, and optimizer. (3) Layer-specific metadata is used to enhance the information basis of the specific layers, e.g., its keywords and parameters. In the following, we will describe these three components of the FAIRnets Ontology correspondingly.[5]

General information describe general components of the neural network, as well as the intended use. For instance, the owner/developer of the (trained) neural network is modeled by using the property `dc:creator`. This attribute makes it possible to search for repositories by the author in the domain of neural networks. Following the Linked Data Principles, the author is represented via a URI. In this way, the authors are uniquely identified. Therefore, it is possible to link it to the Microsoft Academic Knowledge Graph [Fär19] which models scholarly data such as scientific publications in which some of the represented neural network models are proposed and evaluated. Moreover, a name (`rdfs:label`) and a description (`dc:description`) of the trained neural network are stored. The data property `nno:dataset` of type URI allows us to specify the data set that was used to train the neural network. This information already gives a more detailed insight into the neural network as well as its intended use.

Furthermore, the timestamp of creation date (`dc:created`) or last modification (`dc:modified`) allows assessing the currency of the neural network. `dc:license` indicates the rights to modify and redistribute that network. Besides, the property `nno:hasRepositoryLink` allows linking to the repository in which the neural net-

[5] We will use `nno` as the prefix for the namespace `https://w3id.org/nno/ontology#`

work is located. Likewise, references to published papers can be included using `dc:references`.

Model-specific information covers model-specific components of the neural network, such as optimization function denoted by `nno:hasOptimizer`. The ontology covers modeling various loss functions, such as binary cross-entropy and mean squared error, via the property `nno:hasLossFunction`. Loss functions are subdivided into classification and regression loss functions in the ontology to further indicate the intended use of the neural network. The information about existing layers of the neural network can be linked via the property `nno:hasLayer`. The loss functions and layer types available in Keras, an open-source deep learning framework to model neural networks, served as a basis to model available loss functions and layers.

Layer-specific metadata outlines additional information about the individual layer. The layers of neural networks are subdivided into subclasses such as core, recurrent, and convolutional layers. These classes are further subdivided into more specific layer classes. This specification derived from Keras enables the categorization of the neural networks. For example, a neural network with a layer from class convolutional layer can be assigned to the type convolutional neural network. Furthermore, the hyperparameters (e.g., kernel size, stride, and padding) are denoted by `nno:hasLayerKeywords` and saved as a dictionary. Additional values in the layer are denoted by `nno:hasLayerParameter`.

Most of the categories, properties, and instances are annotated with a label (`rdfs:label`), a description (`rdfs:comment`), and, if given, a link (`rdfs:seeAlso`) which make it easy for ontology users to identify the intended use of categories, properties, and instances, therefore supporting the reusability.

6.3.2 Provisioning

The World Wide Web Consortium (W3C) Permanent Identifier Community Group service is used to provide secure and permanent URL forwarding to the ontology. The FAIRnets Ontology in syntax turtle is accessible under https://w3id.org/nno/ontology. Moreover, the ontology has been registered at LOV under https://lov.linkeddata.es/dataset/lov/vocabs/nno. The ontology is licensed under Creative Commons BY 4.0 [Com] which allows its wide usage. Furthermore, the ontology follows the 5-Star Linked Data Principles [Ber] and can, therefore, be easily reused. A VoID file is provided under https://w3id.org/nno/fairnetsvoid including provisioning information.

6.4 FAIRnets Knowledge Graph

Apart from the ontology, we provide the FAIRNETS knowledge graph, which is based on the FAIRNETS ONTOLOGY. The knowledge graph allows us to store knowledge (in our case, detailed metadata for neural network models) intuitively as a graph. Existing and widely used W3C standards and recommendations, such as RDF and SPARQL, can be used to query the knowledge graph and to integrate relatively easily into existing frameworks and systems. For instance, FAIRNETS is already integrated into KBox [MSBC17] which is a data management framework, allowing users to share resources among different applications.

6.4.1 Creation Process

The previous online available neural network repositories such as Keras [Ker], Caffe Model Zoo [BAI], and Wolfram Alpha [Alp] are rather small (under one hundred neural networks) and not sufficient to present trends in the development and usage of neural networks. General-purpose online code-sharing services, such as GitHub [Git] and Bitbucket [Bit], in contrast, contain many repositories of different nature. We, thus, decided to use GitHub since it is the largest host of repositories. Details about the nontrivial extraction process are given in the following.

Data Source. We extract and represent metadata of publicly available, trained neural network models in RDF* (i.e. RDF and RDFS) based on the FAIRNETS ONTOLOGY. Information from SemanGit [KBG19] and GHTorrent [Gou13] can be used to identify GitHub repositories. SemanGit and GHTorrent provide a collection of data extracted from GitHub. In total there are more than 119 million repositories available in the GHTorrent data collection. However, SemanGit and GHTorrent have a different focus and do not provide all the information which we wanted to provide in the FAIRNETS knowledge graph. For instance, information about the architectures of neural networks within the repositories, the creation date of the repositories, as well as the watcher count is not included. We, therefore, directly accessed the GitHub Repository API and queried available neural network repositories. We used the search term ‘neural network’ and filtered for repositories that use Python as a programming language. We accessed these repositories[6] and extracted the neural network metadata.

Extraction Process. The difficulty lies in the extraction of the architecture information from the code. We narrowed our extraction down on neural networks implemented in Python. Still, it is difficult to identify the Python file which models a neural network. Therefore, we started with h5 files which are an open-source

[6] Exemplary GitHub API Request: https://api.github.com/repos/dmnelson/sentiment-analysis-imdb, last acc. 2020-10-15.

Table 6.1: Mapping GitHub REST API values to general components in FAIRNETS.

GitHub API	FAIRnets Ontology
created_at	`dc:created`
description, readme	`dc:description`
html_url	`nno:hasRepositoryLink`
license	`dc:license`
owner [‘html_url’]	`dc:creator`
updated_at	`dc:modified`
watchers_count	`nno:stars`
name	`rdfs:label`
topics [‘names’]	`doap:category`

technology for storing trained machine learning models. Neural networks that have been trained with Keras, for example, can be stored in this format. The h5 file contains information about the neural network, e.g., the sequence of layers, used activation functions, optimization function, and loss function. Accessing the information in the h5 file makes it easier to identify and extract the architecture of the neural network. However, not every repository contains trained neural networks in h5 files. The reason is that trained neural networks often take up a lot of storage space. Thus, our contribution is the information extraction from the code directly which will be described below.

General Information: The mapping of the values from the Github API with the corresponding general component properties in FAIRNETS can be seen in Tab. 6.1. We use the *full_name* of the GitHub REST API as a unique identifier (e.g., ‘dmnelson/sentiment-analysis-imdb’ in note 6.4.1). The *full_name* consists of the GitHub username combined with the name of the repository. The owner of the repository is also the owner of the neural network. Moreover, we store the link (`nno:hasRepositoryLink`), the time of creation (`dc:created`), and the last modification (`dc:modified`) of the repository. As a description of the neural network (`dc:description`), we extracted automatically the description and readme file of the GitHub repository. This gives a summary of the possible use of the neural network. Furthermore, license information about the neural network is extracted and modeled in the knowledge graph, if available. This information is crucial regarding the reusability of neural networks. Given this information, it is possible to filter neural networks by license – which is often an important constraint in industrial settings. To enrich the knowledge graph FAIRNETS with information according to the usage of a neural network, we extract the topics[7] of each repository from the GitHub repositories and store them as `doap:category`.

Additionally, we extract arXiv HTTP links within the readme file and map them to `dc:references`. If BibTex file codes can be found in the readme file, we extract the URL information from the BibTex entry and link it by using the property `dc:references`. The property `dc:references` is only intended for scientific contributions. By linking it with URLs from BibTex entries and arXiv links, we ensure this condition. Other links in the readme file are linked to the neural network using `rdfs:seeAlso`.

Model & Technical Information: The main feature of FAIRnets is the modeling of neural networks. We can model the structure and technical components of neural networks by employing the FAIRnets Ontology. To extract the neural network information from the repositories we consider all Python files in the repositories. Each repository can contain several models of a neural network. In general, it is difficult to extract the architecture information automatically without executing the source code. By executing the code, you can save the neural network model, for example in h5, and retrieve the information easier. We seek a more elegant way by saving execution costs and use language processing to extract the information. Due to that, we focus on Python files with static variables. Despite this restriction, there are still challenges because of various programming styles such as inconsistent naming of variables, complex loop constructions, different structures of code, and other logic statements. Another challenge is changing parameter naming due to different framework versions which are usually not stated. To solve these tasks, a general method is generated using Python Abstract Syntax Trees (AST) module [Pyt]. The AST module helps Python applications process trees of the Python abstract syntax grammar. We focused on Keras applications of neural networks to extract the architecture because it is the most used deep learning framework among the top-winners on Kaggle [Ker]. The information on the architecture of the neural network is then modeled by using the schema and properties provided by the FAIRnets Ontology. Also, the individual layers and their hyperparameters are stored in our knowledge graph. Likewise, the used optimization function and loss function are stored, among other things, allowing us to infer whether the neural network is used for classification or regression. Our code can be found on GitHub under https://github.com/annugyen/FAIRnets.

Evaluation. To evaluate the accuracy of our information extraction, we manually went through 50 examples where we judged the extraction of the GitHub Repository API in Tab. 6.1. The evaluation was in all cases correct. In the case of the neural network architecture, we used the h5 files, if available, in the repositories. We were able to evaluate over 1,343 h5 files with architecture information (i.e., layer

[7] https://developer.github.com/v3/repos/#list-all-topics-for-a-repository, last acc. 2020-10-15.

Table 6.2: Statistical key figures about FAIRNETS.

Key Figure	Value
repositories	9,516
unique users	8,637
neural networks	18,463
FFNN	8,924 (48%)
CNN	6,667 (36%)
RNN	2,872 (16%)

information) that overlap with the architecture extracted from the code with 54% accuracy. Due to later modifications in the code, the overlap with the h5 file does not apply anymore (e.g., if a layer is commented out).

6.4.2 Provisioning

Just like the FAIRNETS ONTOLOGY, the knowledge graph FAIRNETS is also based on the 5-Star Linked Data Principles. The knowledge graph is accessible under a persistent URI from w3id and additionally provided on Zenodo. In combining FAIR principles and Linked Data Principles using URIs to identify things, providing information using RDF*, and linking to other URIs, it is possible to easily reference and use FAIRNETS (see Section 6.7). Machine-readable metadata allows us to describe and search for neural networks. The knowledge graph FAIRNETS, like the ONTOLOGY, is published under the Creative Commons BY 4.0 [Com] license. A VoID file describing the knowledge graph in a machine-readable format is provided under https://w3id.org/nno/fairnetsvoid.

6.4.3 Statistical Analysis of the FAIRnets Knowledge Graph

Tab. 6.2 shows some key figures about the created knowledge graph. It consists of 18,463 neural networks, retrieved from 9,516 repositories, and provided by 8,637 unique users. The creation time of the neural networks in our knowledge graph ranges from January 2015 to June 2019. All these networks have a link to the respective repository and owner. Based on the used layers, we can infer the type of neural network. If a network uses a convolutional layer, it is inferred that the network is a convolutional neural network (CNN). Likewise, if a network contains a recurrent layer, it is inferred that the network is a recurrent neural network (RNN). For simplicity, if none of those two layer types are used, the default claim for the network is a feed-forward neural network (FFNN). Of the total 18,463 neural networks, FFNN is most represented in the knowledge graph comprising half of the neural networks. CNNs follow with 36% and RNN with 16% of the total number of neural networks.

6.5 FAIR Principles for Neural Networks

With FAIRNets, we treat neural networks as research data. As such, to ensure good scientific practice, it should be provided according to the FAIR principles, that is, the data should be findable, accessible, interoperable, and reusable. While the GitHub repositories themselves do not satisfy the FAIR principles (e.g., the metadata is not easily searchable and processable by machines), the modeling of the neural networks in the FAIRNets knowledge graph is made *FAIR* as we show in the following. Specifically, in this section, we identify the factors that make the neural network representations in FAIRNets *FAIR*. This was achieved by following the *FAIRification process* [FAI]. Our FAIRification process is aligned with the *FAIRMetrics* [Wil+18] outlined in Tab. 6.3. In the following, we point out how the single FAIR metrics are met by our knowledge graph.
Findable describes the property that metadata for digital assets is easy for both humans and machines to find. Our approach ensured that, firstly, by retrieving the metadata available in the repository, secondly, structuring its metadata in the readme file, and thirdly, obtaining the architecture information from the code file according to the FAIRNets Ontology. The neural networks we model have unique identifiers (i.e., fulfilling *Gen2_FM_F1A*) and a persistent URI (*Gen2_FM_F1B*). As a result, the process for a human to find a suitable neural network through resource identifiers in the metadata (*Gen2_FM_F3*) is improved. By using RDF as the data model and by providing a schema in OWL as well as a VoID file as a description of the knowledge graph, the metadata is machine-readable (*Gen2_FM_F2*). Thus, the knowledge graph can be automatically filtered and used by services. An exemplary service supporting this statement is presented in Section 6.6. FAIRNets allows for querying information about and within the architecture of the neural networks which was not possible previously. Now, complex queries are feasible (e.g., list all recurrent neural networks published in 2018), which cannot be solved by traditional keyword searches. The metric *Gen2_FM_F4*[8] – ‘indexed in a searchable resource’ – was not passed by FAIRNets although we indexed it on Zenodo. The reason is that the resource on Zenodo is not findable in the search engine Bing which the authors of the FAIRMetrics use as ground truth. However, FAIRNets is indexed by the search engine Google.

[8] https://github.com/FAIRMetrics/Metrics/blob/master/FM_F4, last acc. 2020-10-15

Table 6.3: Evaluation of FAIRNETS according to the Generation2 FAIRMetrics (Note: ✓ = passed, (✓) = should pass, ✗ = not passed).

Principle	FAIRMetric	Name	Result
	Gen2_FM_F1A	Identifier Uniqueness	✓
	Gen2_FM_F1B	Identifier persistence	✓
Findable	*Gen2_FM_F2*	Machine-readability of metadata	✓
	Gen2_FM_F3	Resource Identifier in Metadata	✓
	Gen2_FM_F4	Indexed in a searchable resource	✗
	Gen2_FM_A1.1	Access Protocol	✓
	Gen2_FM_A1.2	Access authorization	✓
Accessible	*Gen2_FM_A2*	Metadata Longevity	(✓)
	Gen2_FM_I1	Use a Knowledge Representation Language	✓
Interoperable	*Gen2_FM_I2*	Use FAIR Vocabularies	✓
	Gen2_FM_I3	Use Qualified References	✓
	Gen2_FM_R1.1	Accessible Usage License	✓
Reusable	*Gen2_FM_R1.2*	Detailed Provenance	(✓)
	Gen2_FM_R1.3	Meets Community Standards	(✓)

Accessible describes that users can access (meta)data using a standardized communication protocol. The protocol must be open, free, and universally implemented. FAIRNETS ONTOLOGY AND KNOWLEDGE GRAPH is located on a web server and can be accessed using the HTTPS protocol (*Gen2_FM_A1.1*). The neural networks in the repositories can also be accessed using the HTTPS protocol (*Gen2_FM_A1.2*). In addition to the open protocol, the accessible property requires that metadata can be retrieved, even if the actual digital assets are no longer available. Due to the separation of the information in FAIRNETS and the actual neural networks on GitHub, this property is fulfilled, since the information in FAIRNETS is preserved even if the neural networks on GitHub are no longer available (*Gen2_FM_A2*). The service to evaluate the metric *Gen2_FM_A2* – 'metadata longevity' – could not be executed because it only tests files that are less than 300kb[9] whereas FAIRNETS has more than 80MB. This test checks for the existence of the 'persistence policy' predicate. This predicate is available in FAIRNETS, which should pass the test.
Interoperable refers to the capability of being integrated with other data as well as being available to applications for analysis, storage, and further processing. We make use of Linked Data by applying RDF (*Gen2_FM_I1*) and SPARQL to represent the information. This makes the data machine-readable, even without the specification of an ad-hoc algorithm or mapping. Additionally, the FAIRNETS ONTOLOGY and the respective KNOWLEDGE GRAPH use well-established and commonly used vocabularies to represent the information. Among others, Dublin Core, Vocabulary of a Friend (VOAF), Creative Commons (CC), and a vocabulary for annotating vocabulary descriptions (VANN) are used for annotations and descriptions (*Gen2_FM_I2*). As a further requirement of the FAIR guideline, qualified references to further metadata are required. This requirement is fulfilled by `rdfs:seeAlso` and `dc:references` (*Gen2_FM_I3*). `dc:references` statements provide scientific references between the neural networks and scientific contributions. These references to the scientific contributions are provided via globally unique and persistent identifiers, such as DOIs.
Reusable aims at achieving well-defined digital assets. This facilitates replicability and usage in other contexts (i.e., reproducibility), as well as findability. Due to the architecture and metadata extraction, the process of finding and reusing a neural network by an end-user becomes significantly easier and can now be performed systematically. By using best practices in ontology building, the properties and classes of FAIRNETS ONTOLOGY provided are self-explanatory with labels and descriptions (*Gen2_FM_R1.3*). The neural networks in FAIRNETS contain structured detailed metadata such as creator and GitHub link (see *Gen2_FM_R1.2*) for easy findability and reuse. At the same time, most neural networks in FAIRNETS have an assigned license which is important for reusability (*Gen2_FM_R1.1*). For passing *Gen2_FM_R1.2*, (meta)data must be associated with detailed provenance reusing existing vocabularies such as Dublin Core which we included in our knowledge graph. *Gen2_FM_R1.3* tests

[9] https://github.com/FAIRMetrics/Metrics/blob/master/FM_A2, last acc. 2020-10-15

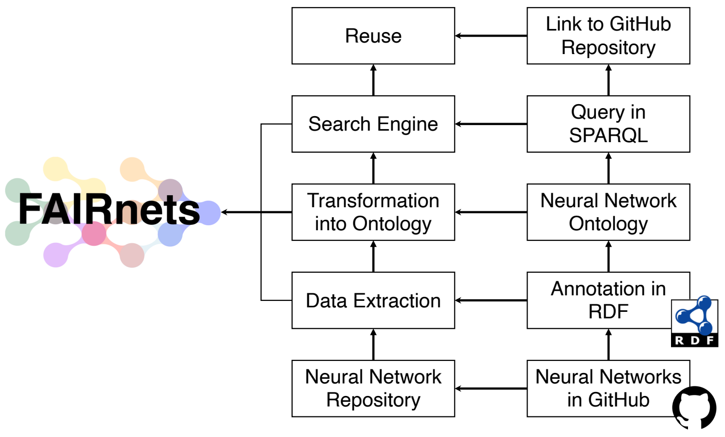

Reuse
Link to GitHub Repository
Search Engine
Query in SPARQL
FAIRnets
Transformation into Ontology
Neural Network Ontology
Data Extraction
Annotation in RDF
RDF
Neural Network Repository
Neural Networks in GitHub

Figure 6.2: The FAIRNETS SEARCH Framework

a certification saying that the resource is compliant with a minimum of metadata. FAIRNETS is described by using LOV standards for publication. Therefore, we assume that these metrics are fulfilled. Overall, the neural networks modeled in FAIRNETS fulfill all requirements of the FAIR principles, see Tab. 6.3.

6.6 FAIRnets Search

FAIRNETS SEARCH is a service provided by us to make neural networks searchable and findable. The service is available at the following URI http://km.aifb.kit.edu/services/fairnets/. It represents an attempt to search for all neural network architectures or neural network instances that fulfill specific requirements (e.g., used for specific tasks, having a specific architecture, etc.). For this purpose, the Web service uses our knowledge graph FAIRNETS. This knowledge graph currently contains 18.463 neural networks. Figure 6.2 shows an overview of the framework. We collected neural networks from GitHub and retrieved the information. The data is annotated using the FAIRNETS ONTOLOGY and represented in RDF. For each of the neural networks in FAIRNETS, the relevant properties according to the Neural Network Ontology such as the description and the architecture are stored. That way, FAIRNETS can be queried with a set of desired properties and responds with a set of neural networks that have these properties with a *SPARQL Endpoint*. The FAIRNETS SEARCH combines these implementations by a browser-based frontend to the SPARQL endpoint.

Demonstration of Use Cases

The attendees of the demo will learn how FAIRNETS SEARCH can be used to gain insights into the usage of existing neural networks and data sets in machine learning. In the online demo, the users are encouraged to use the search engine to find and access neural network architectures. With FAIRNETS SEARCH, we will tackle the following three scenarios:

Search for Neural Networks. The FAIRNETS SEARCH engine allows users to search keyword-based for neural networks. FAIRNETS is searched using SPARQL. Multiple keywords are supported in the search. The results are sorted based on the number of hits counted, i.e. how often the keywords appear in the title and description. The attendee of the demo can, for example, search for the terms 'image' and 'classification' and will get a list of neural networks that are related to these terms (see Figure 6.3). Existing neural networks in this area can thus easily be retrieved. Detailed information on the individual neural networks can be accessed on the model sites of the neural network. Information such as the publisher, links, architecture information, and the latest update of the network is provided and shown by our demo. The attendees of the demo can choose based on the information and links provided by us if an already modeled neural network fits their use case. We support the reusability of neural networks with FAIRNETS SEARCH.

Search for Used Data Sets. Another use case is the usage of data sets. Attendees of the demo can search for specific data sets (e.g. search for 'mnist'). FAIRNETS SEARCH lists neural networks that are related to the searched data set. This gives the attendees the possibility to find out which neural network architectures have been applied to a given data set. Additional information such as the link to the GitHub repository is available on the respective pages. This allows for getting more information about the performance of the architectures on the data sets. Besides identifying already applied neural network architectures on a given data set, the search can also be used to identify new data sets. This information is implicit in the descriptions of neural networks. Searching for 'image classification' lists all available neural networks in this domain. In the description of the neural network or on the corresponding GitHub repository page further information about the used data sets for training can be found. This supports the attendees of the demo to find new data sets suitable for their use case.

Fine-grained Search by Exploiting the SPARQL Endpoint. Besides the search functionality, we offer the attendees of the demo the possibility to post individual SPARQL queries to the FAIRNETS endpoint. We use YASGUI [RH17] to display

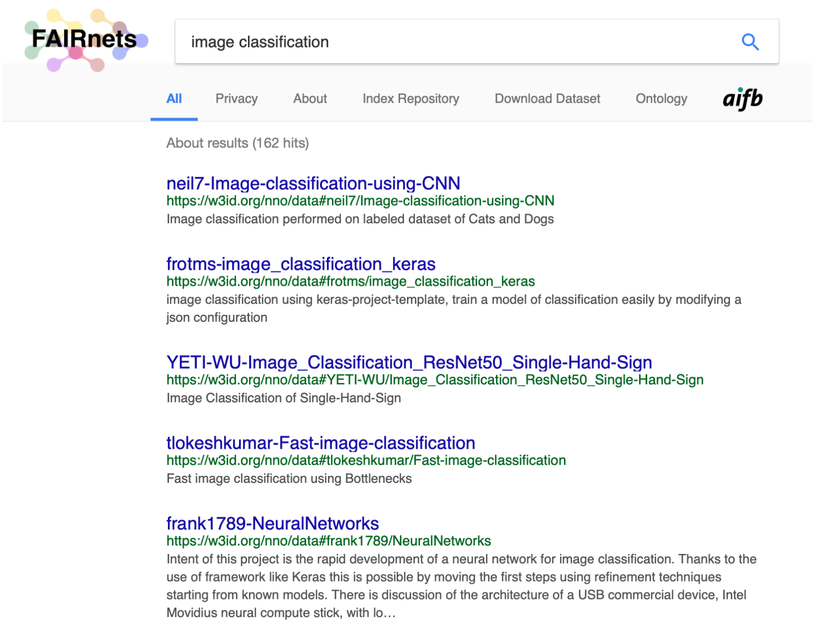


image classification

About results (162 hits)

neil7-Image-classification-using-CNN
https://w3id.org/nno/data#neil7/Image-classification-using-CNN
Image classification performed on labeled dataset of Cats and Dogs

frotms-image_classification_keras
https://w3id.org/nno/data#frotms/image_classification_keras
image classification using keras-project-template, train a model of classification easily by modifying a json configuration

YETI-WU-Image_Classification_ResNet50_Single-Hand-Sign
https://w3id.org/nno/data#YETI-WU/Image_Classification_ResNet50_Single-Hand-Sign
Image Classification of Single-Hand-Sign

tlokeshkumar-Fast-image-classification
https://w3id.org/nno/data#tlokeshkumar/Fast-image-classification
Fast image classification using Bottlenecks

frank1789-NeuralNetworks
https://w3id.org/nno/data#frank1789/NeuralNetworks
Intent of this project is the rapid development of a neural network for image classification. Thanks to the use of framework like Keras this is possible by moving the first steps using refinement techniques starting from known models. There is discussion of the architecture of a USB commercial device, Intel Movidius neural compute stick, with lo…

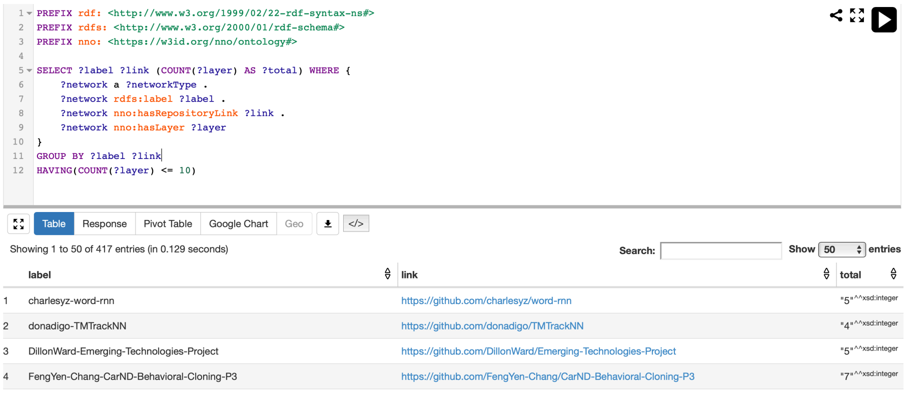
Figure 6.3: Returned hits of the FAIRNETS SEARCH demo, based on the entered keywords. In the search presented here, the user was interested in image classification.

the results of the queries. The interface to the provided endpoint can be accessed via the following link: https://km.aifb.kit.edu/services/fairnets/sparql. The endpoint allows for answering individual requests upon the data set. We already offer some pre-selected SPARQL queries, such as a list of all neural networks with a maximum number of layers (see Figure 6.4) and an overview of the frequencies of the activation functions used.

6.7 Impact

We see high potential of FAIRNETS ONTOLOGY and the knowledge graph FAIRNETS in the areas of *transparency*, *recommendation*, *reusability*, *education*, and *search*. In the following, we outline these application areas in more detail.

Transparency. Neural networks are applied in many different areas such as finance [QJLQ20], medical health [Kha+01], and law [PWB00]. Transparency plays a major



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX nno: <https://w3id.org/nno/ontology#>

SELECT ?label ?link (COUNT(?layer) AS ?total) WHERE {
    ?network a ?networkType .
    ?network rdfs:label ?label .
    ?network nno:hasRepositoryLink ?link .
    ?network nno:hasLayer ?layer
}
GROUP BY ?label ?link
HAVING(COUNT(?layer) <= 10)
```

Showing 1 to 50 of 417 entries (in 0.129 seconds)

Search: Show 50 entries

	label	link	total
1	charlesyz-word-rnn	https://github.com/charlesyz/word-rnn	"5"^^xsd:integer
2	donadigo-TMTrackNN	https://github.com/donadigo/TMTrackNN	"4"^^xsd:integer
3	DillonWard-Emerging-Technologies-Project	https://github.com/DillonWard/Emerging-Technologies-Project	"5"^^xsd:integer
4	FengYen-Chang-CarND-Behavioral-Cloning-P3	https://github.com/FengYen-Chang/CarND-Behavioral-Cloning-P3	"7"^^xsd:integer

Figure 6.4: Example of a List of all GitHub links, which provide neural networks that have at most 10 layers.

role in these areas when it comes to trust the output of a used neural network model. We claim that our contribution which makes neural networks more transparent can increase trust and privacy [Sch19]. Additionally, using semantic annotations can even enhance interpretability by distributional semantics [SFH19].

Another aspect is the transparency of scientific work regarding neural networks. Researchers publishing a model should provide it according to the FAIR principles to strengthen their scientific contribution. Our knowledge graph FAIRNETS can pave the way for this.

Recommendation. Neural Architecture Search (NAS) is used to find the best suitable neural network architecture based on existing architectures [EMH19]. However, the search is performed purely based on metrics like accuracy ignoring explainability aspects concerning the best fitting model. Our knowledge graph allows us to have a search for the best suitable neural network models on a meta-level, using modeled use-cases, data sets, and scientific papers. Knowledge graphs have also been used to provide explanations for recommendations to the user [Wan+19a, Xia+19].

Additionally, we can apply explainable reasoning [Wan+19b] given the ontology and the knowledge graph and infer some rules. Doing this, we might reason which neural network models are reasonable or which components of the architecture stand in conflict with each other.

Reusability. Transfer learning is a method in deep learning to reuse pre-trained models on new tasks. Our contribution facilitates the search of pre-trained neural networks and provides the metadata needed to choose a specific neural network. We can envision FAIRNETS linked with other knowledge bases to enrich the reusability of neural networks by applying Linked Data Principles [Ber]. For example, training data sets can be linked with Neural Data Server [YAF20], Wikidata [Wik], and Zenodo [Zen] through schema.org [Sch], scientific papers can be linked with the Microsoft Academic Knowledge Graph [Fär19], and metadata can be extended with OpenAIRE [Ope].

On the other hand, providing a model and encouraging its reuse can improve it by revealing limitations, errors, or suggestions to other tasks.

Education. Our FAIRNETS knowledge graph can be used for educational purposes [Che+18], for instance, to learn best practices regarding designing a neural network model. Another aspect is to learn the usages of different architectures and their approaches (e.g., via linked papers). Our knowledge graph includes training parameters that can help setting up the training process of a neural network (e.g., when facing the cold start problem).

Search. We provide online the search system FAIRNETS SEARCH [NW19], which is based on the proposed FAIRNETS ONTOLOGY and KNOWLEDGE GRAPH. Users can search for neural network models through search terms. Additional information can be retrieved by using SPARQL as query language on top of our knowledge graph, which enables faceted and semantic search capabilities. The SPARQL endpoint is also available to the public. The search system shows how a semantic search system can be realized which improved the limited capabilities of keyword searches on GitHub. Furthermore, developers can provide their GitHub repository to run the FAIRification process on their neural networks. Until now, we have over 550 visits to the website FAIRNETS SEARCH with over 4,800 page views, 1,400 searches on our website with an average duration of twelve minutes, and the maximal actions in one visit is 356.

6.8 Conclusion

This chapter was dedicated to making neural networks FAIR. To this end, we first proposed the FAIRNETS ONTOLOGY, an ontology that allows us to model neural networks on a fine-grained level and that is easily extensible. Second, we provided the knowledge graph FAIRNETS. This graph contains rich metadata of 18,463 publicly available neural network models using our proposed ontology as knowledge schema. Third, we provide FAIRNETS SEARCH which allows for making neural networks better findable, searchable, and accessible which enhances efficiency. We have shown high potential impact in transparency, recommendation, reusability, education, and search.

If we look at the research questions again, we can draw the following conclusions. The information encoded using our ontology facilitates transparency by following fair principles. This answers Research Question 4.1. As we retrieved the architecture information from the source code, we have the necessary parameters to apply an existing model to a novel use case which answers Research Question 4.2. Regarding Research Question 4.3, we modeled three information levels, namely general, model, and layer-specific information, that enhance the reusability.

Outlook. As future work, we plan to connect the FAIRNETS ONTOLOGY and KNOWLEDGE GRAPH with scholarly data. Specifically, we will work on linking publications, authors, and venues modeled in knowledge graphs like the Microsoft Academic Knowledge Graph or Wikidata to the FAIRNETS knowledge graph. This will require applying sophisticated information extraction methods to scientific publications.

7 Conclusion

Neural networks are used in all kinds of industries but behave like a black box. To apply and improve these black boxes, explanations are crucial. As neural networks are also used by non-computer scientists, it is becoming increasingly important that humans can understand and improve them. Furthermore, companies have to deal with the issue of data protection, which is reinforced by politics (GDPR). We, therefore, addressed the following question in this thesis:

Research Question. *How to make explanations human-understandable?*

In this thesis, we provided a framework for neural networks consisting of different approaches. In line with state-of-the-art research, our framework tackles important goals to make explanations human-understandable. These are transparency, scrutability, trust, effectiveness, persuasiveness, efficiency, and satisfaction. We contributed TransPer, ObAlEx, FilTag and FAIRnets to achieve these goals. These are summarized in the following by recapitulating the hypotheses with their associated research questions and concluding with the findings, limitations, and outlooks.

In Chapter 3, we introduced TransPer in the context of product recommendation in e-commerce. We showed how explanations based on LRP can be quantified to get human-understandable and beneficial explanations according to our individuality, certainty, and diversity measure. Although we worked with RNNs, this approach applies to all neural network types with even heterogeneous data as input.

Hypothesis 1. *Quantifying explanations based on the relevance of the input features facilitates the evaluation of not only the input data but also the neural network.*

1.1 What parameters are relevant for understanding explanation quality?

1.2 Which implications can be derived with these relevancies?

1.3 Can these relevancies be used to improve the neural network (and if so, how)?

Research Question 1.1. addresses the relevant parameters to understand the explanation. We found out that based on the LRP approach, the weights and also bias is relevant to understanding the explanation quality. Most importantly, the bias is usually omitted from the literature, although we could show that it reveals a lot in connection with the explanation. Research Question 1.2. addresses implications that can be derived. The explanation quality measures reveal how important specific relevancies are for different customer bases. Research Question 1.3 addresses the improvement of a neural network based on these explanations. The leave-one-out method can reveal important features which can be used to improve a neural network. In summary, we confirm Hypothesis 1. We explained fluctuations in the prediction qualities. This helped in finding ideas to improve the neural network and understand the customer base of online shops. Although we have only shown our approach in e-commerce, our metrics can generally be applied to all neural network types, as LRP is also applicable to other architectures.

In Chapter 4, we improved existing visual explanation methods by quantifying the explanations following humans' expectations in ObAlEx. We have shown for CNNs in image classification that our explanation quality score can be used to improve a classifier regarding object-aligned explanations. Training a model with the addition of the data of the desired explanation increases not only accuracy but also effectiveness and scrutability. This leads to more generalized models and therefore more trust in a model.

Hypothesis 2. *In image classification, object-aligned explanations can sufficiently map the desired explanation of humans and guarantee an intuitive explanation.*

2.1 How can right for the right reasons be measured?

2.2 Does it satisfy the concept of classifying right for the right reasons?

2.3 Can it be included in the training process?

Research Question 2.1 addresses the selection of the desired explanation regarding classifying an image right for the right reasons. The ObAlEx metric measures the alignment of the explanations with the actual object and thus can be used to measure the right features for the right reasons. Research Question 2.2 addresses the sufficiency of the metric regarding the concept of classifying right for the right reasons. By considering different state-of-the-art explanation methods in our approach, the

ObAlEx metric satisfies the concept. Research Question 2.3 addresses the included in the training process. The ObAlEx metric can be calculated during the training process to see the improvement of the explanations in addition to accuracy and loss function. In summary, we confirm Hypothesis 2. Although ObAlEx is not explicitly integrated into the loss function but included as an additional step, it facilitates more generalized models which can increase the user's trust in the model by object-aligned explanations.

In Chapter 5, we provided FilTag a semantic description of filters in a CNN to make them human-understandable. Also here, we have shown in the context of CNNs in image classification that filters encode specific information that can be translated into a semantic description. These can be better understood by humans and computers than visual explanations which lead to more scrutability and trust.

Hypothesis 3. *Filters in CNNs encode semantic information of the input images.*

3.1 *How can the encoded semantic information of the filters be decoded?*

3.2 *How precise are the tagged filters in predicting and understanding the output of the CNN (compared to visual methods)?*

3.3 *What benefit does link tagged filters to knowledge graphs offer?*

Research Question 3.1 addresses the decoding of semantic information in convolutional filters. We could show that by tagging the filters of convolutional layers, the encoded semantic information of the filters can be decoded in human-understandable language. Research Question 3.2 addresses the tagged filters and their information value regarding prediction and understanding the output. We found out that words can be more comprehensible compared to visual methods which are currently used more and more often. We could show with our evaluation that the tagged filters are quite accurate in predicting and understanding the output of the CNN. Research Question 3.3 addresses the benefit of linking these tags to knowledge graphs. Linking the tagged filters to other knowledge graphs allows to expand the information of the labeled filters and with it more knowledge, which can lead to more understanding. In summary, we confirm Hypothesis 3. Our approach allows for explicit, non-visual explanations in contrast to state-of-the-art explanations which are more understandable for non-experts.

Finally, in Chapter 6, we provided a collection and preparation of publicly available neural networks following the fair principles. Based on this, we created a knowledge graph called FAIRnets with metadata over 18,000 neural networks. This knowledge graph based on FAIRnets Ontology allows the presentation of neural network metadata in a transparent way. We have additionally integrated this data in FAIRnets Search to enhance efficiency in findability, search, and accessibility.

Hypothesis 4. *Neural network models which comply with the FAIR Principles support humans in dealing with the increase in volume, complexity, and creation speed of these models.*

4.1 Which information should be provided by the neural network models to enable transparency?

4.2 Which information must be provided to apply an existing model to a novel use case?

4.3 What kind of functions should be provided for supporting humans in reusing neural networks?

Research Question 4.1 addresses the information basis necessary to facilitate transparency. We could show that the information encoded using our ontology enhances transparency by following fair principles. In providing it this way, the neural networks themselves become research data. Research Question 4.2 addresses the information basis necessary to reproduce a model. As we retrieved the architecture information from the source code, we have the necessary parameters to reproduce the model. Research Question 4.3 addresses the features to enhance reusability. We modeled three information levels, namely general, model, and layer-specific information, that enhance the reusability. In summary, we confirm Hypothesis 4. We have shown high potential impact in transparency, recommendation, reusability, education, and search even though additional information extraction methods could enhance the information basis.

All in all, our framework expands along two explanation dimensions. First, we go into the depth of explanation with our first three approaches and look at specific components of the neural network. These approaches allow us to zoom into a specific neural network. These are ObAlEx, FilTag and TransPer. They not only provide a quantified explanation but also point out ideas to improve the neural network. We have complemented these approaches with another approach, namely FAIRnets, that allows us to provide a broader explanation of neural networks. Using semantics, we have developed an ontology that provides an overview of existing neural networks as an explanation. To the best of our knowledge, such a broad approach to explaining neural networks has not been proposed before. Coming back to the main research question, we could show that following the goals of explanation mentioned in Section 1.2 give a good starting point to make explanations human-understandable. In the course of this thesis, we found that certain targets are more prominent than others. For example, *trust* could be covered in three of the four approaches which is related to the fact that FAIRnets focuses on *transparency* and *efficiency* by providing metadata on neural networks. It does not address the processing of the data which would foster trust. Instead, facts are presented in a structured way and the perception of the user is not taken into account. This approach shows that the commitment to some goals and their optimization can diminish other goals. TransPer, on the other hand, addresses *transparency* but is more accessible to user's sensitivity. Due to its design in the context of recommender systems, it is more open to the needs of a user, whose goals are *persuasiveness* and *satisfaction* among others. ObAlEx and FilTag consider besides *trust* also *scrutability*. This is possible because these approaches explain the inner workings of the model. Consequently, with the help of these approaches, we were able to understand the networks through study and observation. Conversely, the goal *effectiveness* is only in ObAlEx because it uses multiple explanatory approaches and thus gives enough room to adjust the explanation.

Outlook

Following the seven goals of explanations, it was possible to design a framework that tackles all goals. However, in this work, not all goals were treated the same. As mentioned before, trust is considered in three approaches (namely ObAlEx, FilTag, and TransPer) but satisfaction and persuasiveness was only considered in TransPer which has a more important role in the field of recommender systems. By optimizing one goal, the other can be comprised or ignored. As future work, we can work on how to align these goals, e.g. by more user-friendly interfaces, so that all targets are addressed.

In this thesis, we only showed each approach on one neural network repository, neural network type, input data, or use-case. For example, FAIRnets only uses GitHub as a neural network repository to create a knowledge graph. This can be extended to other repositories. We did not address it due to different standards, frameworks, and programming language which would require more engineering but would not further demonstrate the benefits of our approach. To enrich the information base further information extraction methods would be needed. ObAlEx is only applied to one neural network type, namely CNNs. This can be extended on other neural networks such as RNNs with sequential data. We did not approach it because we used object detection to automatically get the desired explanation. Using other neural network types requires other methods or manual labeling of the desired explanation. FilTag is only applied on images as input data. This can also be extended with other labeled input data. Transper is only shown in the context of e-commerce. This can be extended to other businesses using neural networks. We did not tackle it, because of the lack of companies who are willing to share their architecture and data.

Bibliography

[ACÖG18] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. **Towards better understanding of gradient-based attribution methods for Deep Neural Networks**. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.
Cited on pages 27, 29, 40.

[Alb+19] Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. **iNNvestigate Neural Networks!** In *J. Mach. Learn. Res.* volume 20, pages 93:1–93:8, 2019. URL: http://jmlr.org/papers/v20/18-540.html.
Cited on page 5.

[Alp] Wolfram Alpha. **Neural Net Repository of Neural Network Models**. URL: https://resources.wolframcloud.com/NeuralNetRepository.
Cited on pages 74, 78.

[AW93] Norman P. Archer and Shouhong Wang. **Application of the Back Propagation Neural Network Algorithm with Monotonicity Constraints for Two-Group Classification Problems**. In *Decision Sciences* volume 24:1, pages 60–75, 1993.
Cited on pages 5, 21.

[Bac+15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. **On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation**. In *PloS one* volume 10:7, e0130140, 2015.
Cited on pages 6, 23, 27, 28, 29, 40.

[BAI] BAIR. **Caffe Model Zoo**. URL: https://bit.ly/39m0RVT.
Cited on pages 74, 78.

[Bau+17] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. **Network Dissection: Quantifying Interpretability of Deep Visual Representations**. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3319–3327, 2017. DOI: 10.1109/CVPR.2017.354.
Cited on page 59.

[BC17] Or Biran and Courtenay Cotton. **Explanation and justification in machine learning: A survey**. In *IJCAI-17 workshop on explainable AI (XAI)*. Volume 8 of, pages 8–13, 2017.
Cited on page 3.

[Ber] Tim Berners-Lee. **Linked Data**. URL: https://bit.ly/39iQ8eW.
Cited on pages 74, 77, 89.

[BG97] Zvi Boger and Hugo Guterman. **Knowledge extraction from artificial neural network models**. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*. Volume 4 of, pages 3030–3035, 1997.
Cited on page 73.

[Bha18] Homanga Bharadhwaj. **Layer-Wise Relevance Propagation for Explainable Deep Learning Based Speech Recognition**. In *2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Louisville, KY, USA, December 6-8, 2018*, pages 168–174, 2018. DOI: 10.1109/ISSPIT.2018.8642691.
Cited on page 27.

[BHZC20] Elaine M. Bettaney, Stephen R. Hardwick, Odysseas Zisimopoulos, and Benjamin Paul Chamberlain. **Fashion Outfit Generation for E-Commerce**. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part V*. Volume 12461 of Lecture Notes in Computer Science, pages 339–354, 2020. DOI: 10.1007/978-3-030-67670-4_21.
Cited on page 26.

[Bit] Bitbuckt. **Bitbucket**. URL: https://www.bitbucket.org.
Cited on page 78.

[BMRS16] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. **A Neural Click Model for Web Search**. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 531–541, 2016. DOI: 10.1145/2872427.2883033.
Cited on page 26.

[CEP20] Gabriel Dias Cantareira, Elham Etemad, and Fernando V. Paulovich. **Exploring Neural Network Hidden Layer Activity Using Vector Fields**. In *Inf.* volume 11:9, page 426, 2020. DOI: 10.3390/info11090426.
Cited on page 21.

[Che+18] Penghe Chen, Yu Lu, Vincent W. Zheng, Xiyang Chen, and Boda Yang. **KnowEdu: A System to Construct Knowledge Graph for Education**. In *IEEE Access* volume 6, pages 31553–31563, 2018. DOI: 10.1109/ACCESS.2018.2839607.
Cited on page 89.

[Che+19] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. **POG: Personalized Outfit Generation for Fashion Recommendation at Alibaba iFashion**. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2662–2670, 2019. DOI: 10.1145/3292500.3330652.
Cited on page 26.

[CL17] Mingang Chen and Pan Liu. **Performance evaluation of recommender systems**. In *International Journal of Performability Engineering* volume 13:8, page 1246, 2017.
Cited on page 35.

[Com] Creative Commons. **Attribution 4.0 International (CC BY 4.0)**. URL: https://creativecommons.org/licenses/by/4.0/.
Cited on pages 77, 81.

[CSHB18] Aditya Chattopadhyay, Anirban Sarkar, Prantik Howlader, and Vineeth N. Balasubramanian. **Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks**. In *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pages 839–847, 2018. DOI: 10.1109/WACV.2018.00097.
Cited on pages 23, 50.

[DPGK19] Ranjeet Devarakonda, Giri Prakash, Kavya Guntupally, and Jitendra Kumar. **Big Federal Data Centers Implementing FAIR Data Principles: ARM Data Center Example**. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, pages 6033–6036, 2019. DOI: 10.1109/BigData47090.2019.9006051.
Cited on pages 8, 72.

[EBCV09] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. **Visualizing higher-layer features of a deep network**. In *University of Montreal* volume 1341:3, page 1, 2009.
Cited on pages 61, 64, 65.

[eco] econda. **Personalization & Analytics**. URL: https://www.econda.de.
Cited on page 37.

[EMH19] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. **Neural Architecture Search: A Survey**. In *J. Mach. Learn. Res.* volume 20, pages 55:1–55:21, 2019.
Cited on page 88.

[FAI] Go FAIR. **FAIRification Process**. URL: https://bit.ly/3hQdA7L.
Cited on page 82.

[Fär19] Michael Färber. **The Microsoft Academic Knowledge Graph: A Linked Data Source with 8 Billion Triples of Scholarly Data**. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*. Volume 11779 of Lecture Notes in Computer Science, pages 113–129, 2019. DOI: 10.1007/978-3-030-30796-7_8.
Cited on pages 76, 89.

[FFP07] Li Fei-Fei, Robert Fergus, and Pietro Perona. **Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories**. In *Comput. Vis. Image Underst.* volume 106:1, pages 59–70, 2007. DOI: 10.1016/j.cviu.2005.09.012.
Cited on page 51.

[Fou] The Linux Foundation. **ONNX Homepage**. URL: https://onnx.ai.
Cited on page 74.

[FV18] Ruth Fong and Andrea Vedaldi. **Net2Vec: Quantifying and Explaining How Concepts Are Encoded by Filters in Deep Neural Networks**. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8730–8738, 2018. DOI: 10.1109/CVPR.2018.00910.
Cited on page 59.

[GD18] Nick Gisolfi and Artur Dubrawski. **Revealing Actionable Simplicity in Data**. In *2018 AAAI Spring Symposia, Stanford University, Palo Alto, California, USA, March 26-28, 2018*, 2018.
Cited on page 21.

[GDDM14] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. **Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation**. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 580–587, 2014. DOI: 10.1109/CVPR.2014.81.
Cited on page 59.

[GHMS92] Rodney M. Goodman, Charles M. Higgins, John W. Miller, and Padhraic Smyth. **Rule-Based Neural Networks for Classification and Probability Estimation**. In *Neural Comput.* volume 4:6, pages 781–804, 1992. DOI: 10.1162/neco.1992.4.6.781.
Cited on pages 5, 21.

[Git] GitHub. **GitHub**. URL: https://www.github.com.
Cited on page 78.

[Gou13] Georgios Gousios. **The GHTorent dataset and tool suite**. In *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 233–236, 2013.
Cited on page 78.

[GP09] Aldo Gangemi and Valentina Presutti. **Ontology Design Patterns**. In *Handbook on Ontologies*. International Handbooks on Information Systems. 2009, pages 221–243. DOI: 10.1007/978-3-540-92673-3_10.
Cited on page 74.

[Gro] Data Mining Group. **PMML 4.0**. URL: https://bit.ly/3CtQAU1.
Cited on page 74.

[GV16] Gebrekirstos G. Gebremeskel and Arjen P. de Vries. **Recommender Systems Evaluations : Offline, Online, Time and A/A Test**. In *Working Notes of CLEF 2016 - Conference and Labs of the Evaluation forum, Évora, Portugal, 5-8 September, 2016*. Volume 1609 of CEUR Workshop Proceedings, pages 642–656, 2016.
Cited on page 35.

[Hag] Daniel Hagenmayer. **Exploring Convolutional Neural Networks - Labelling of filters**. URL: https://aifb.kit.edu/web/Thema4594.
Cited on page 58.

[Has] Muneeb ul Hassan. **VGG16 – Convolutional Network for Classification and Detection**. URL: https://bit.ly/2Z1z23o.
Cited on page 18.

[HGDG20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. **Mask R-CNN**. In *IEEE Trans. Pattern Anal. Mach. Intell.* volume 42:2, pages 386–397, 2020. DOI: 10.1109/TPAMI.2018.2844175.
Cited on page 51.

[How+17] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. **MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications**. In *CoRR* volume abs/1704.04861, 2017. arXiv: 1704.04861.
Cited on pages 20, 21, 50.

[HPRC20] Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng (Polo) Chau. **Summit: Scaling Deep Learning Interpretability by Visualizing Activation and Attribution Summarizations**. In *IEEE Trans. Vis. Comput. Graph.* volume 26:1, pages 1096–1106, 2020. DOI: 10.1109/TVCG.2019.2934659.
Cited on pages 59, 62, 63.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. **Deep Residual Learning for Image Recognition**. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. DOI: 10.1109/CVPR.2016.90.
Cited on pages 18, 19, 50.

[JLC18] Sen Jia, Thomas Lansdall-Welfare, and Nello Cristianini. **Right for the Right Reason: Training Agnostic Networks**. In *Advances in Intelligent Data Analysis XVII - 17th International Symposium, IDA 2018, 's-Hertogenbosch, The Netherlands, October 24-26, 2018, Proceedings*. Volume 11191 of Lecture Notes in Computer Science, pages 164–174, 2018. DOI: 10.1007/978-3-030-01768-2_14.
Cited on page 48.

[KBG19] Dennis Oliver Kubitza, Matthias Böckmann, and Damien Graux. **SemanGit: A Linked Dataset from git**. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*. Volume 11779 of Lecture Notes in Computer Science, pages 215–228, 2019. DOI: 10.1007/978-3-030-30796-7_14.
Cited on page 78.

[KDS17] K. G. Kapanova, Ivan Tomov Dimov, and Jean Michel D. Sellier. **A neural network sensitivity analysis in the presence of random fluctuations**. In *Neurocomputing* volume 224, pages 177–183, 2017. DOI: 10.1016/j.neucom.2016.10.060.
Cited on page 21.

[Ker] Keras. **Keras Applications**. URL: https://keras.io/applications/.
Cited on pages 17, 18, 74, 78, 80.

[Kha+01] Javed Khan, Jun S Wei, Markus Ringner, Lao H Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R Antonescu, Carsten Peterson, et al. **Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks**. In *Nature medicine* volume 7:6, pages 673–679, 2001.
Cited on page 87.

[KKMK20] Keijiro Kanda, Muthu Subash Kavitha, Jun'ichi Miyao, and Takio Kurita. **Analysis of Information Flow in Hidden Layers of the Trained Neural Network by Canonical Correlation Analysis**. In *Frontiers of Computer Vision - 26th International Workshop, IW-FCV 2020, Ibusuki, Kagoshima, Japan, February 20-22, 2020, Revised Selected Papers*. Volume 1212 of Communications in Computer and Information Science, pages 206–220. Springer, 2020. DOI: 10.1007/978-981-15-4818-5_16.
Cited on page 21.

[KTL20] Mohamed Khoali, Abdelhak Tali, and Yassin Laaziz. **Advanced Recommendation Systems Through Deep Learning**. In *NISS 2020: The 3rd International Conference on Networking, Information Systems & Security, Marrakech, Morocco, March 31 - April 2, 2020*, pages 51:1–51:8, 2020. DOI: 10.1145/3386723.3387870.
Cited on page 26.

[Lap+19] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. **Unmasking Clever Hans Predictors and Assessing What Machines Really Learn**. In *CoRR* volume abs/1902.10178, 2019. arXiv: 1902.10178.
Cited on pages 2, 3.

[LC99] Brian Lees and Juan M. Corchado. **Integrated Case-Based Neural Network Approach to Problem Solving**. In *XPS-99: Knowledge-Based Systems - Survey and Future Directions, 5th Biannual German Conference on Knowledge-Based Systems, Würzburg, Germany, March 3-5, 1999, Proceedings.* Volume 1570 of Lecture Notes in Computer Science, pages 157–166, 1999. DOI: 10.1007/10703016_11.
Cited on pages 5, 21.

[LCZL17] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. **Mining Fashion Outfit Composition Using an End-to-End Deep Learning Approach on Set Data**. In *IEEE Trans. Multim.* volume 19:8, pages 1946–1955, 2017. DOI: 10.1109/TMM.2017.2690144.
Cited on page 26.

[LL17] Scott M. Lundberg and Su-In Lee. **A Unified Approach to Interpreting Model Predictions**. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4765–4774, 2017.
Cited on page 21.

[LM20] Katrien Laenen and Marie-Francine Moens. **A Comparative Study of Outfit Recommendation Methods with a Focus on Attention-based Fusion**. In *Inf. Process. Manag.* volume 57:6, page 102316, 2020. DOI: 10.1016/j.ipm.2020.102316.
Cited on page 26.

[LMS18] Ana L. D. Loureiro, Vera L. Miguéis, and Lucas F. M. da Silva. **Exploring the use of deep neural networks for sales forecasting in fashion retail**. In *Decis. Support Syst.* volume 114, pages 81–93, 2018. DOI: 10.1016/j.dss.2018.08.010.
Cited on page 26.

[Mit+19] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. **Model Cards for Model Reporting**. In *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, Atlanta, GA, USA, January 29-31, 2019*, pages 220–229, 2019. DOI: 10.1145/3287560.3287596.
Cited on pages 73, 76.

[Mon+19] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. **Layer-Wise Relevance Propagation: An Overview**. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Volume 11700. Lecture Notes in Computer Science. 2019, pages 193–209. DOI: 10.1007/978-3-030-28954-6_10.
Cited on pages 6, 23, 27, 29, 40.

[MSBC17] Edgard Marx, Tommaso Soru, Ciro Baron, and Sandro Athaide Coelho. **KBox: Distributing Ready-to-Query RDF Knowledge Graphs**. In *The Semantic Web: ESWC 2017 Satellite Events - ESWC 2017 Satellite Events, Portorož, Slovenia, May 28 - June 1, 2017, Revised Selected Papers*. Volume 10577 of Lecture Notes in Computer Science, pages 54–58, 2017. DOI: 10.1007/978-3-319-70407-4_11.
Cited on page 78.

[MSM18] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. **Methods for interpreting and understanding deep neural networks**. In *Digit. Signal Process.* volume 73, pages 1–15, 2018. DOI: 10.1016/j.dsp.2017.10.011.
Cited on pages 6, 23, 27, 40, 59.

[Mus15] Mark A. Musen. **The protégé project: a look back and a look forward**. In *AI Matters* volume 1:4, pages 4–12, 2015. DOI: 10.1145/2757001.2757003.
Cited on page 74.

[ND19] Naresh Nelaturi and G Devi. **A product recommendation model based on recurrent neural network**. In *Journal Européen des Systèmes Automatisés* volume 52, pages 501–507, 2019.
Cited on page 26.

[NHH15] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. **Learning Deconvolution Network for Semantic Segmentation**. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1520–1528, 2015. DOI: 10.1109/ICCV.2015.178.
Cited on pages 23, 26.

[NHWF21] Anna Nguyen, Daniel Hagenmayer, Tobias Weller, and Michael Färber. **Explaining Convolutional Neural Networks by Tagging Filters**. In *CoRR* volume abs/2109.09389, 2021. arXiv: 2109.09389.
Cited on pages 17, 58.

[NKHF21] Anna Nguyen, Franz Krause, Daniel Hagenmayer, and Michael Färber. **Quantifying Explanations of Neural Networks in E-Commerce Based on LRP**. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track - European Conference, ECML PKDD 2021, Bilbao, Spain, September 13-17, 2021, Proceedings, Part V*. Volume 12979 of Lecture Notes in Computer Science, pages 251–267, 2021. DOI: 10.1007/978-3-030-86517-7_16.
Cited on pages 15, 23, 26.

[NOF21] Anna Nguyen, Adrian Oberföll, and Michael Färber. **Right for the Right Reasons: Making Image Classification Intuitively Explainable**. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part II*. Volume 12657 of Lecture Notes in Computer Science, pages 327–333. Springer, 2021. DOI: 10.1007/978-3-030-72240-1_32.
Cited on pages 26, 47.

[NW19] Anna Nguyen and Tobias Weller. **FAIRnets Search - A Prototype Search Service to Find Neural Networks**. In *Proceedings of the Posters and Demo Track of the 15th International Conference on Semantic Systems co-located with 15th International Conference on Semantic Systems (SEMANTiCS 2019), Karlsruhe, Germany, September 9th - to - 12th, 2019*. Volume 2451 of CEUR Workshop Proceedings. CEUR-WS.org, 2019.
Cited on pages 72, 89.

[NWFS20] Anna Nguyen, Tobias Weller, Michael Färber, and York Sure-Vetter. **Making Neural Networks FAIR**. In *Knowledge Graphs and Semantic Web - Second Iberoamerican Conference and First Indo-American Conference, KGSWC 2020, Mérida, Mexico, November 26-27, 2020, Proceedings*. Volume 1232 of Communications in Computer and Information Science, pages 29–44. Springer, 2020. DOI: 10.1007/978-3-030-65384-2_3.
Cited on pages 64, 72.

[Obe] Adrian Oberföll. **Quantifying the Explanations of Convolutional Neural Networks**. URL: https://aifb.kit.edu/web/Thema42762.
Cited on page 47.

[Ola+18] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. **The building blocks of interpretability**. In *Distill* volume 3:3, e10, 2018.
Cited on page 59.

[OMS17] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. **Feature visualization**. In *Distill* volume 2:11, e7, 2017.
Cited on pages 59, 61, 64, 65.

[Ope] OpenAIRE. **OpenAIRE**. URL: https://www.openaire.eu.
Cited on page 89.

[Par00] Sungjune Park. **Neural Networks and Customer Grouping in E-Commerce: A Framework Using Fuzzy ART**. In *2000 Academia / Industry Working Conference on Research Challenges (AIWoRC 2000), Next Generation Enterprises: Virtual Organizations and Mobile / Pervasive Technologies, 27-29 April 2000, Buffalo, NY, USA*, pages 331–336, 2000. DOI: 10.1109/AIWORC.2000.843312.
Cited on page 26.

[PE16] European Parliament and Council of the European Union. **Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)**. In, pages 1–88, 2016.
Cited on pages 2, 26.

[PM00] H Sofia Pinto and JP Martins. **Reusing ontologies**. In *AAAI 2000 Spring Symposium on Bringing Knowledge to Business Processes*. Volume 2 of, page 7, 2000.
Cited on page 74.

[PWB00] Susan W Palocsay, Ping Wang, and Robert G Brookshire. **Predicting criminal recidivism using neural networks**. In *Socio-Economic Planning Sciences* volume 34:4, pages 271–284, 2000.
Cited on page 87.

[Pyt] Python. **Abstract Syntax Trees**. URL: https://bit.ly/3klZxIW.
Cited on page 80.

[QJLQ20] Songqiao Qi, Kaijun Jin, Baisong Li, and Yufeng Qian. **The exploration of internet finance by using neural network**. In *J. Comput. Appl. Math.* volume 369, 2020. DOI: 10.1016/j.cam.2019.112630.
Cited on page 87.

[RH17] Laurens Rietveld and Rinke Hoekstra. **The YASGUI family of SPARQL clients**. In *Semantic Web* volume 8:3, pages 373–383, 2017. DOI: 10.3233/SW-150197.
Cited on page 86.

[RHD17] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. **Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations**. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2662–2670, 2017. DOI: 10.24963/ijcai.2017/371.
Cited on pages 48, 51.

[RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. **Learning representations by back-propagating errors**. In *nature* volume 323:6088, pages 533–536, 1986.
Cited on pages 5, 26.

[RSG16] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. **"Why Should I Trust You?": Explaining the Predictions of Any Classifier**. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016. DOI: 10.1145/2939672.2939778.
Cited on pages 1, 2, 5, 21, 22, 48, 50.

[Rus+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. **ImageNet Large Scale Visual Recognition Challenge**. In *Int. J. Comput. Vis.* volume 115:3, pages 211–252, 2015. DOI: 10.1007/s11263-015-0816-y.
Cited on pages 50, 64.

[Sch] Schema. **Schema**. URL: https://schema.org/docs/about.html.
Cited on page 89.

[Sch+20] Patrick Schramowski, Wolfgang Stammer, Stefano Teso, Anna Brugger, Xiaoting Shao, Hans-Georg Luigs, Anne-Katrin Mahlein, and Kristian Kersting. **Right for the Wrong Scientific Reasons: Revising Deep Networks by Interacting with their Explanations**. In *CoRR* volume abs/2001.05371, 2020. arXiv: 2001.05371.
Cited on pages 48, 51.

[SCH17] Robyn Speer, Joshua Chin, and Catherine Havasi. **ConceptNet 5.5: An Open Multilingual Graph of General Knowledge**. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451, 2017.
Cited on pages 64, 65.

[Sch19] Daniel Schwabe. **Trust and Privacy in Knowledge Graphs**. In *Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 722–728, 2019. DOI: 10.1145/3308560.3317705.
Cited on page 88.

[SDBR15] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. **Striving for Simplicity: The All Convolutional Net**. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015. arXiv: 1412.6806.
Cited on pages 6, 23, 26, 59.

[Sel+16] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. **Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization**. In *CoRR* volume abs/1610.02391, 2016. arXiv: 1610.02391.
Cited on pages 23, 48, 50.

[SFH19] Vivian Dos Santos Silva, André Freitas, and Siegfried Handschuh. **On the Semantic Interpretability of Artificial Intelligence Models**. In *CoRR* volume abs/1907.04105, 2019. arXiv: 1907.04105.
Cited on page 88.

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. **Learning Important Features Through Propagating Activation Differences**. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Volume 70 of Proceedings of Machine Learning Research, pages 3145–3153, 2017.
Cited on pages 23, 26.

[SM19] Wojciech Samek and Klaus-Robert Müller. **Towards Explainable Artificial Intelligence**. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Volume 11700. Lecture Notes in Computer Science. 2019, pages 5–22. DOI: 10.1007/978-3-030-28954-6_1.
Cited on page 22.

[Smi+17] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. **SmoothGrad: removing noise by adding noise**. In *CoRR* volume abs/1706.03825, 2017. arXiv: 1706.03825.
Cited on page 6.

[STY17] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. **Axiomatic Attribution for Deep Networks**. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Volume 70 of Proceedings of Machine Learning Research, pages 3319–3328, 2017.
Cited on pages 6, 26.

[SVZ14] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. **Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps**. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014. arXiv: 1312.6034.
Cited on pages 6, 26, 59.

[SZ15] Karen Simonyan and Andrew Zisserman. **Very Deep Convolutional Networks for Large-Scale Image Recognition**. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. arXiv: 1409.1556.
Cited on pages 17, 19, 50, 58, 61.

[SZ19] Hai Shu and Hongtu Zhu. **Sensitivity Analysis of Deep Neural Networks**. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4943–4950, 2019. DOI: 10.1609/aaai.v33i01.33014943.
Cited on page 21.

[Sze+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. **Going deeper with convolutions**. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9. IEEE Computer Society, 2015. DOI: 10.1109/CVPR.2015.7298594.
Cited on page 20.

[Sze+16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. **Rethinking the Inception Architecture for Computer Vision**. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826, 2016. DOI: 10.1109/CVPR.2016.308.
Cited on pages 20, 58.

[TM12] Nava Tintarev and Judith Masthoff. **Evaluating the effectiveness of explanations for recommender systems - Methodological issues and empirical studies on the impact of personalization**. In *User Model. User Adapt. Interact.* volume 22:4-5, pages 399–439, 2012. DOI: 10.1007/s11257-011-9117-5.
Cited on page 4.

[Wan+19a] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. **KGAT: Knowledge Graph Attention Network for Recommendation**. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 950–958, 2019. DOI: 10.1145/3292500.3330989.
Cited on page 88.

[Wan+19b] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. **Explainable Reasoning over Knowledge Graphs for Recommendation**. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5329–5336, 2019. DOI: 10.1609/aaai.v33i01.33015329.
Cited on page 88.

[Wik] Wikidata. **Wikidata**. URL: https://www.wikidata.org.
Cited on page 89.

[Wil+16] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. **The FAIR Guiding Principles for scientific data management and stewardship**. In *Scientific data* volume 3:1, pages 1–9, 2016.
Cited on pages 8, 72.

[Wil+18] Mark D Wilkinson, Susanna-Assunta Sansone, Erik Schultes, Peter Doorn, Luiz Olavo Bonino da Silva Santos, and Michel Dumontier. **A design framework and exemplar metrics for FAIRness**. In *Scientific data* volume 5:1, pages 1–4, 2018.
Cited on page 82.

[Wis+19] John Wise, Alexandra Grebe de Barron, Andrea Splendiani, Beeta Balali-Mood, Drashtti Vasant, Eric Little, Gaspare Mellino, Ian Harrow, Ian Smith, Jan Taubert, et al. **Implementation and relevance of FAIR data principles in biopharmaceutical R&D**. In *Drug discovery today* volume 24:4, pages 933–938, 2019.
Cited on pages 8, 72.

[Wu+18] Mike Wu, Michael C. Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. **Beyond Sparsity: Tree Regularization of Deep Models for Interpretability**. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1670–1678, 2018.
Cited on page 21.

[Xia+19] Yikun Xian, Zuohui Fu, S. Muthukrishnan, Gerard de Melo, and Yongfeng Zhang. **Reinforcement Knowledge Graph Reasoning for Explainable Recommendation**. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 285–294, 2019. DOI: 10.1145/3331184.3331203.
Cited on page 88.

[YAF20] Xi Yan, David Acuna, and Sanja Fidler. **Neural Data Server: A Large-Scale Search Engine for Transfer Learning Data**. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 3892–3901, 2020. DOI: 10.1109/CVPR42600.2020.00395.
Cited on page 89.

[YLLZ10] Jin Yuan, Yanming Li, Chengliang Liu, and Xuan F. Zha. **Leave-One-Out Cross-Validation Based Model Selection for Manifold Regularization**. In *Advances in Neural Networks - ISNN 2010, 7th International Symposium on Neural Networks, ISNN 2010, Shanghai, China, June 6-9, 2010, Proceedings, Part I*. Volume 6063 of Lecture Notes in Computer Science, pages 457–464, 2010. DOI: 10.1007/978-3-642-13278-0_59.
Cited on page 34.

[Zen] Zenodo. **Zenodo**. URL: https://zenodo.org.
Cited on page 89.

[ZF14] Matthew D. Zeiler and Rob Fergus. **Visualizing and Understanding Convolutional Networks**. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*. Volume 8689 of Lecture Notes in Computer Science, pages 818–833, 2014. DOI: 10.1007/978-3-319-10590-1_53.
Cited on pages 5, 22, 48, 50, 59, 62.

[Zho+16] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. **Learning Deep Features for Discriminative Localization**. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2921–2929, 2016. DOI: 10.1109/CVPR.2016.319.
Cited on page 23.

List of Figures

List of Tables

A List of Publications

[1] **Explaining Convolutional Neural Networks by Tagging Filters**. In *CoRR.* Volume abs/2109.09389, arXiv: 2109.09389, 2021. Anna Nguyen, Daniel Hagenmayer, Tobias Weller and Michael Färber.

[2] **Quantifying Explanations of Neural Networks in E-Commerce Based on LRP**. In *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track, ECML PKDD 2021.* Volume 12979, pages 251–267, 2021. Anna Nguyen, Franz Krause, Daniel Hagenmayer and Michael Färber.

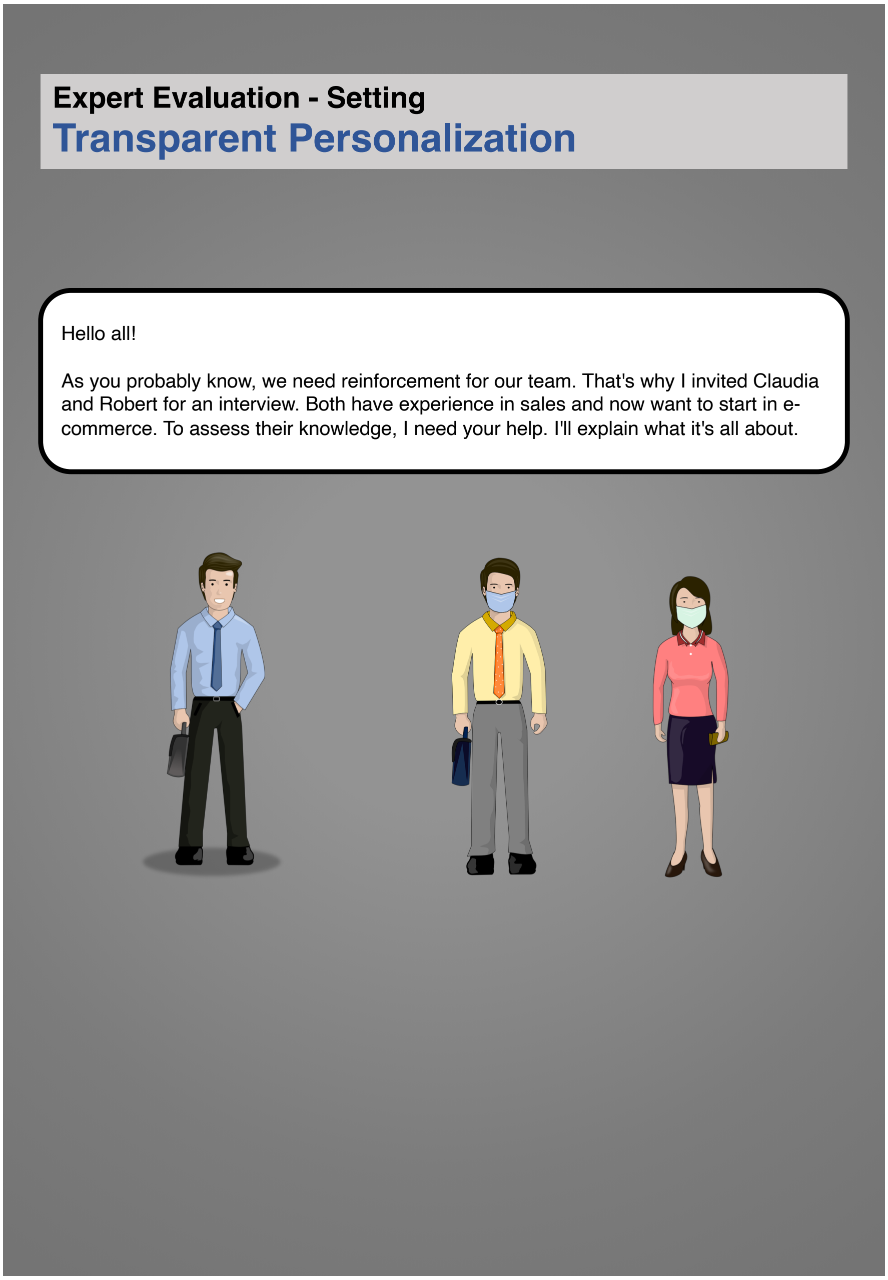
[3] **Right for the Right Reasons: Making Image Classification Intuitively Explainable**. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021.* Volume 12657, pages 327–333, 2021. Anna Nguyen, Adrian Oberföll and Michael Färber.

[4] **Making Neural Networks FAIR**. In *Knowledge Graphs and Semantic Web - Second Iberoamerican Conference and First Indo-American Conference, KGSWC 2020.* Volume 1232, pages 29–44, 2020. Anna Nguyen, Tobias Weller, Färber and York Sure-Vetter.

[5] **FAIRnets Search – A Prototype Search Service to Find Neural Networks**. In *Proceedings of the Posters and Demo Track of the 15th International Conference on Semantic Systems co-located with 15th International Conference on Semantic Systems (SEMANTiCS 2019).* Volume 2451, 2019. Anna Nguyen and Tobias Weller.

B Preliminary Study of Expert Evaluation

This is an exemplary excerpt of our preliminary study of expert evaluation from Chapter 3. This study was conducted at two online stores, each with two and three experts from the corresponding data science area, respectively. The goal was to find out how the study should be conducted in order to design sufficient explanations for further evaluation for end customers.

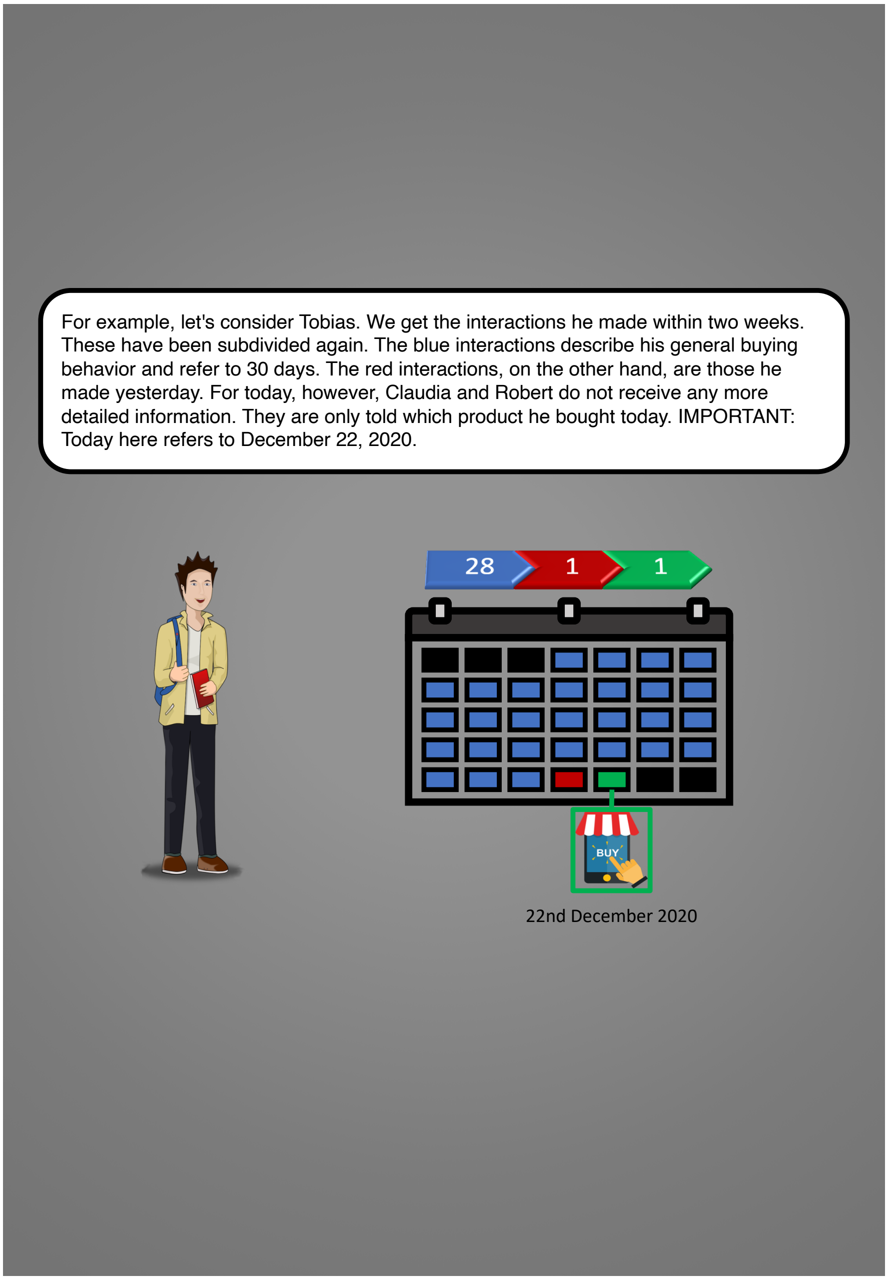


Expert Evaluation - Setting
Transparent Personalization

Hello all!

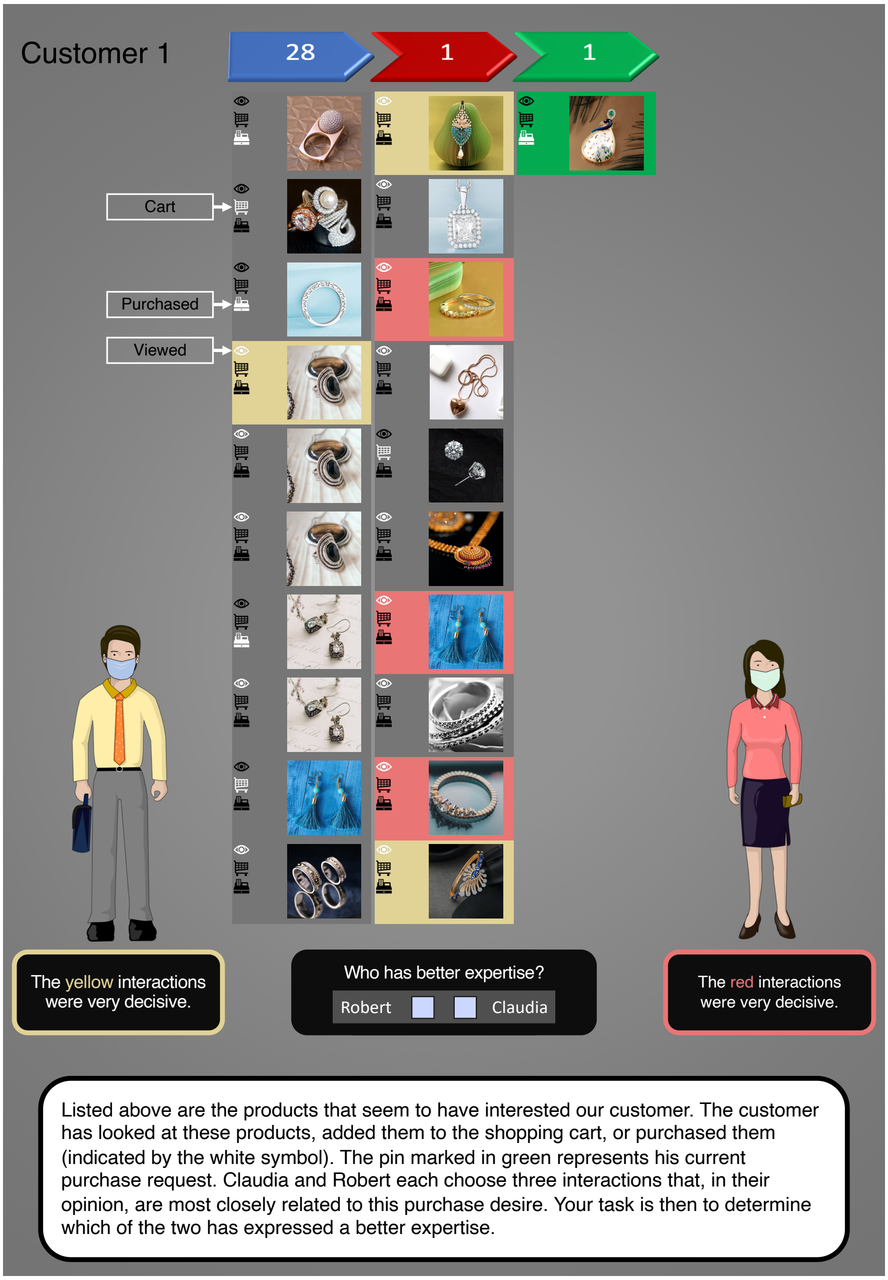
As you probably know, we need reinforcement for our team. That's why I invited Claudia and Robert for an interview. Both have experience in sales and now want to start in e-commerce. To assess their knowledge, I need your help. I'll explain what it's all about.

In the following, we will look at the profile data of some of our customers, which was kindly provided to us. Over the last two weeks, we have recorded them and classified them chronologically.

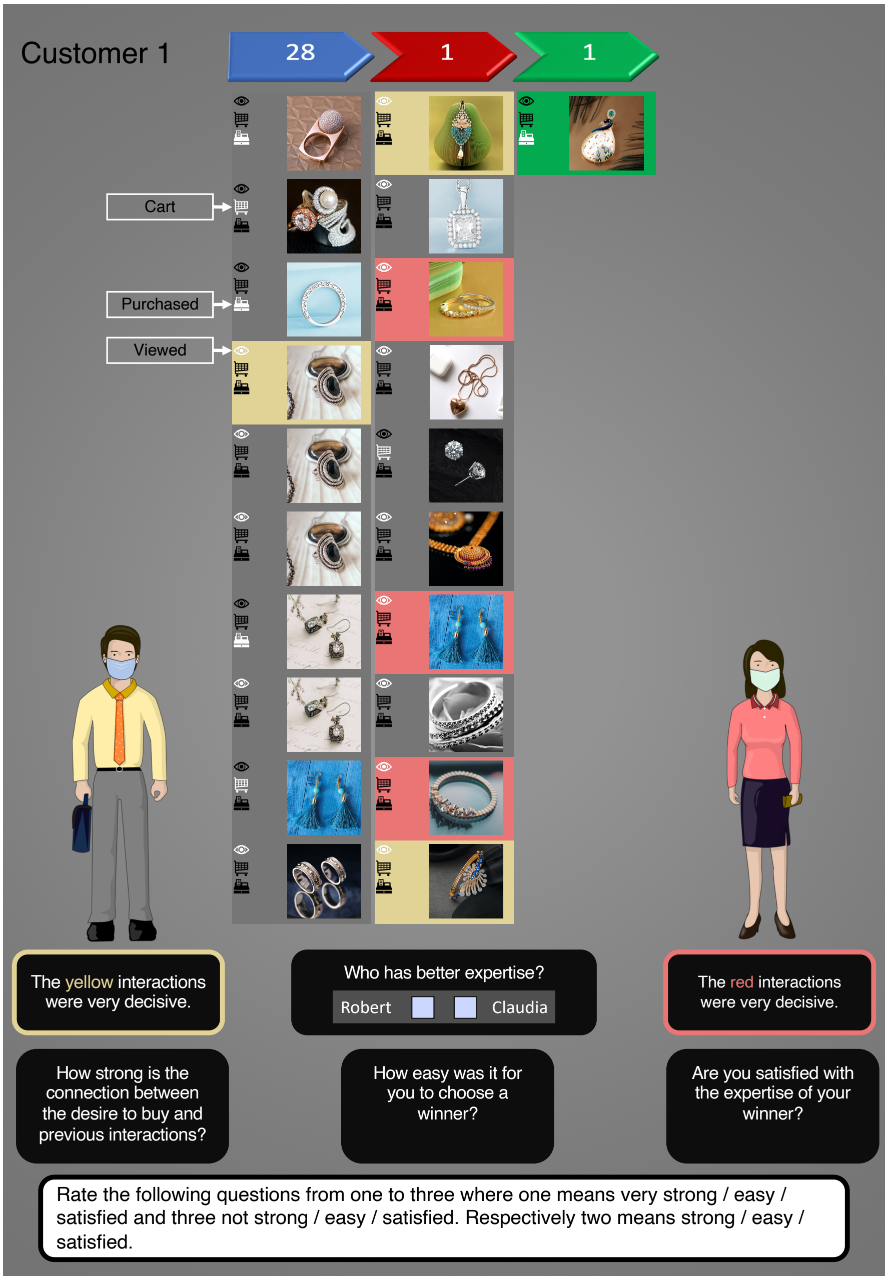


For example, let's consider Tobias. We get the interactions he made within two weeks. These have been subdivided again. The blue interactions describe his general buying behavior and refer to 30 days. The red interactions, on the other hand, are those he made yesterday. For today, however, Claudia and Robert do not receive any more detailed information. They are only told which product he bought today. IMPORTANT: Today here refers to December 22, 2020.

Our customers all think that today's purchase desire is related to their previous interactions. Claudia and Robert are now required to find out what these are. Afterwards, you are to judge who is more likely to be able to interpret the customer's preferences correctly. I will illustrate this with a case study.


Customer 1
28
1
1
Cart
Purchased
Viewed
The yellow interactions were very decisive.
Who has better expertise?
Robert
Claudia
The red interactions were very decisive.

Listed above are the products that seem to have interested our customer. The customer has looked at these products, added them to the shopping cart, or purchased them (indicated by the white symbol). The pin marked in green represents his current purchase request. Claudia and Robert each choose three interactions that, in their opinion, are most closely related to this purchase desire. Your task is then to determine which of the two has expressed a better expertise.



The yellow interactions were very decisive.

Who has better expertise?

Robert ☐ ☐ Claudia

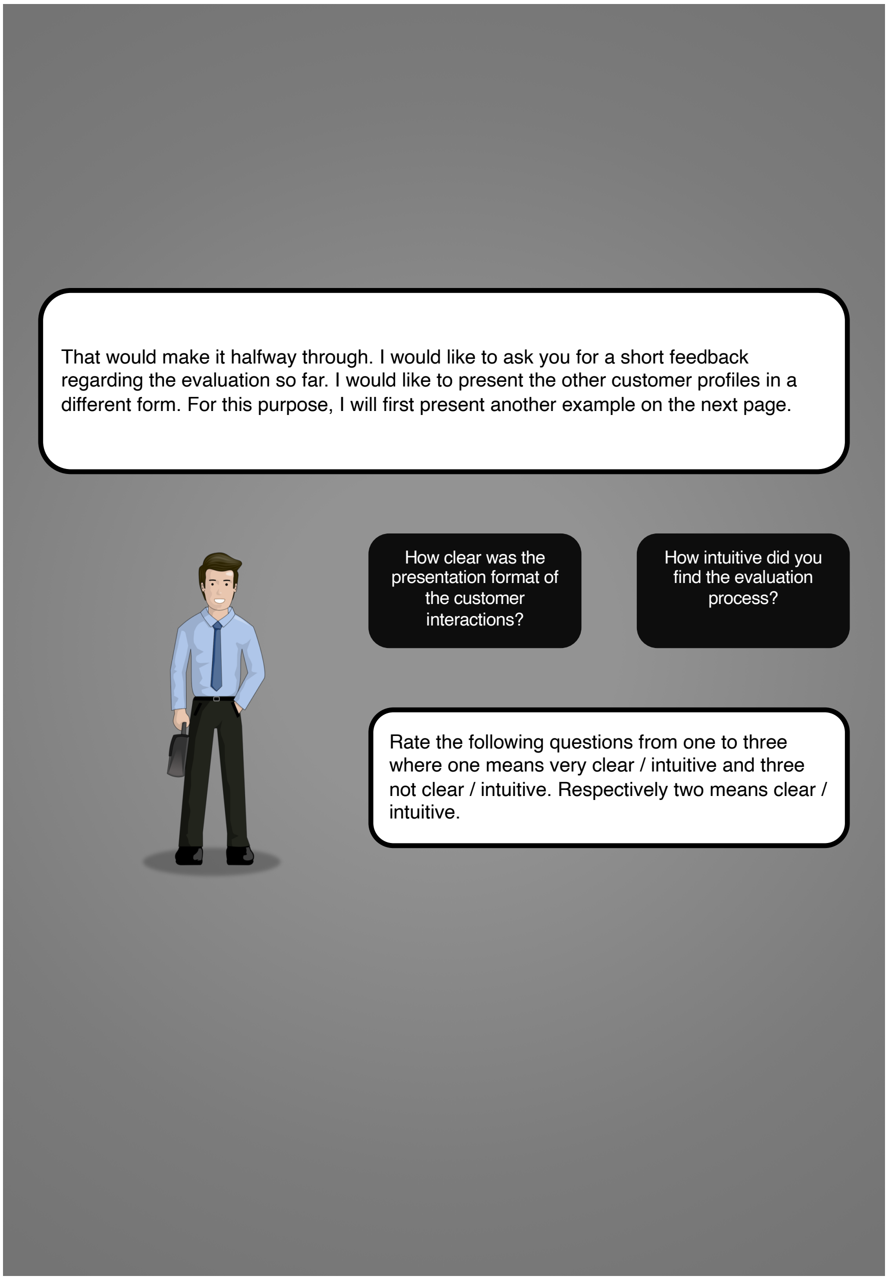
The red interactions were very decisive.

How strong is the connection between the desire to buy and previous interactions?

How easy was it for you to choose a winner?

Are you satisfied with the expertise of your winner?

Rate the following questions from one to three where one means very strong / easy / satisfied and three not strong / easy / satisfied. Respectively two means strong / easy / satisfied.



That would make it halfway through. I would like to ask you for a short feedback regarding the evaluation so far. I would like to present the other customer profiles in a different form. For this purpose, I will first present another example on the next page.

How clear was the presentation format of the customer interactions?

How intuitive did you find the evaluation process?

Rate the following questions from one to three where one means very clear / intuitive and three not clear / intuitive. Respectively two means clear / intuitive.

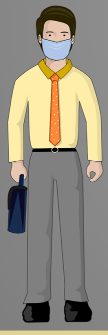
Customer 16

28 ▶ 1 ▶ 1 ▶

Product	Interaction	#
Liebeskind Berlin Earrings LJ-0307-E-11	Purchased	1
Ti Sento - Milano Lady‘s Ring 1936SY/54	Viewed	3
Ti Sento - Milano Lady‘s Ring 1936SY/54	Cart	1
Ti Sento - Milano Lady‘s Ring 1871ZY/56	Viewed	2
Skagen Lady‘s Ring SKJ1271998	Viewed	1
Ti Sento - Milano Lady‘s Ring 1871ZY/56	Cart	1
Ti Sento - Milano Lady‘s Ring 12104SY/54	Viewed	1

Product	Interaction
FAVS Chain 88101294	Viewed
FAVS Chain 88101294	Viewed
FAVS Chain 88101294	Cart
Ti Sento - Milano Lady‘s Ring 1936SY/54	Viewed
FAVS Chain 88101294	Purchased

Product	Interaction
Engelsrufer Lady‘s Ring Shiny ERR-SHINY-ZI-56	Purchased



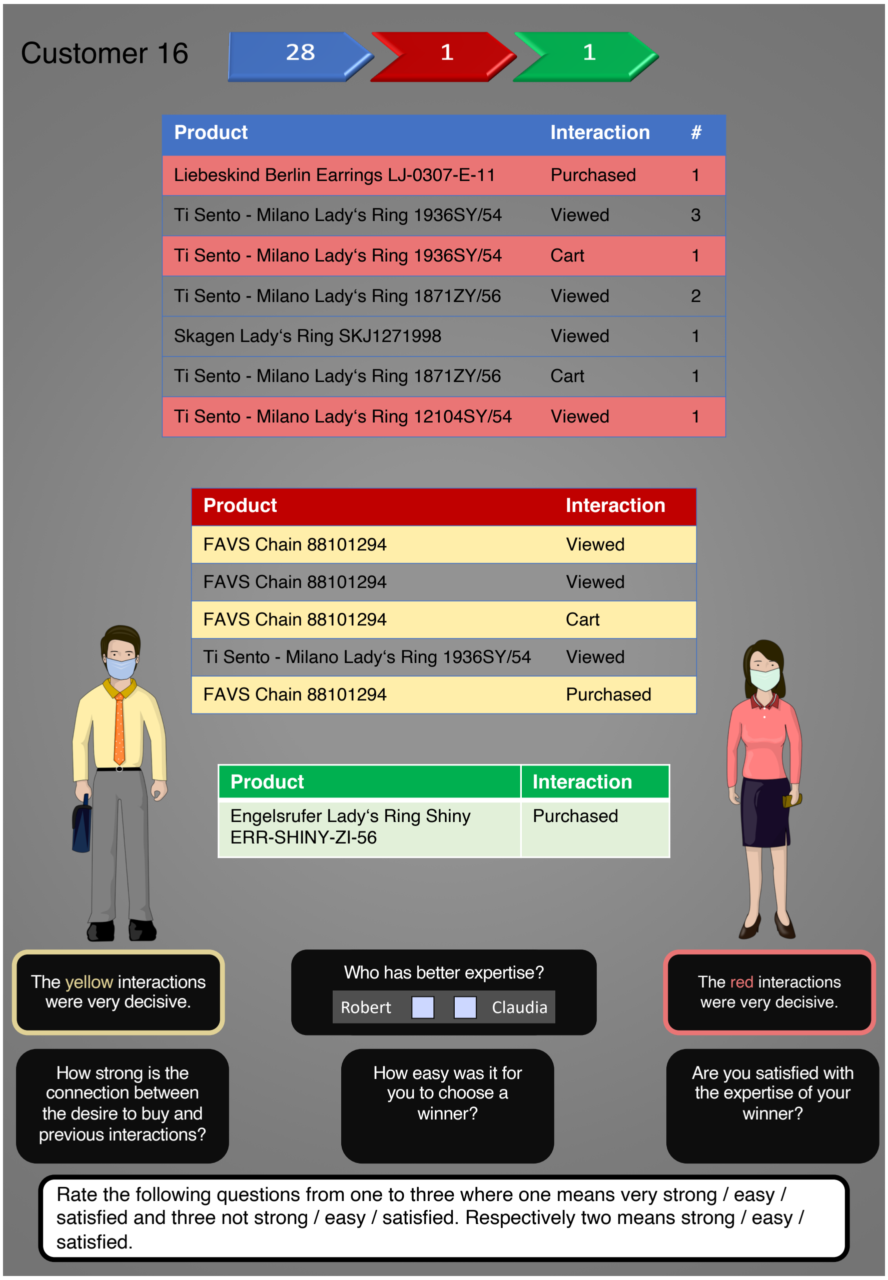
The yellow interactions were very decisive.

Who has better expertise?

Robert ☐ ☐ Claudia

The red interactions were very decisive.

Listed above are the products that seem to have interested our customer. The customer has looked at these products, added them to the shopping cart, or purchased them. Blue interactions can occur more frequently (see #). Red interactions, on the other hand, are considered separately and are ordered chronologically. The product highlighted in green represents his current purchase desire. Claudia and Robert each choose three interactions that are most closely related to this purchase desire. Your task is then to determine which of the two has expressed a better expertise.



Customer 16

28 → 1 → 1

Product	Interaction	#
Liebeskind Berlin Earrings LJ-0307-E-11	Purchased	1
Ti Sento - Milano Lady‘s Ring 1936SY/54	Viewed	3
Ti Sento - Milano Lady‘s Ring 1936SY/54	Cart	1
Ti Sento - Milano Lady‘s Ring 1871ZY/56	Viewed	2
Skagen Lady‘s Ring SKJ1271998	Viewed	1
Ti Sento - Milano Lady‘s Ring 1871ZY/56	Cart	1
Ti Sento - Milano Lady‘s Ring 12104SY/54	Viewed	1

Product	Interaction
FAVS Chain 88101294	Viewed
FAVS Chain 88101294	Viewed
FAVS Chain 88101294	Cart
Ti Sento - Milano Lady‘s Ring 1936SY/54	Viewed
FAVS Chain 88101294	Purchased

Product	Interaction
Engelsrufer Lady‘s Ring Shiny ERR-SHINY-ZI-56	Purchased

The yellow interactions were very decisive.

Who has better expertise?

Robert ☐ ☐ Claudia

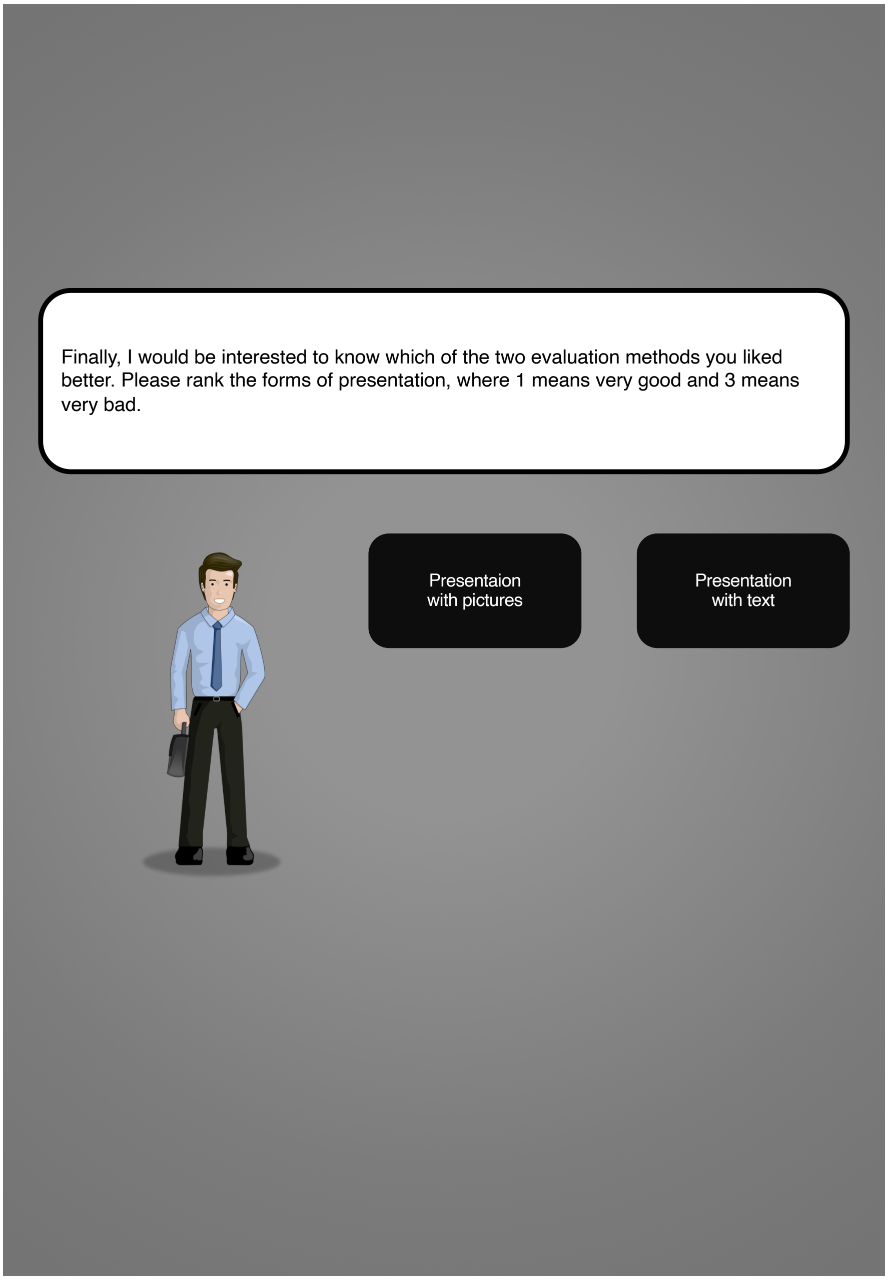
The red interactions were very decisive.

How strong is the connection between the desire to buy and previous interactions?

How easy was it for you to choose a winner?

Are you satisfied with the expertise of your winner?

Rate the following questions from one to three where one means very strong / easy / satisfied and three not strong / easy / satisfied. Respectively two means strong / easy / satisfied.



Finally, I would be interested to know which of the two evaluation methods you liked better. Please rank the forms of presentation, where 1 means very good and 3 means very bad.