Opening the Machine Learning Black Box with Layer-wise Relevance Propagation

vorgelegt von M. Sc.

SEBASTIAN LAPUSCHKIN GEB. BACH

Von der Fakultät IV – Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Benjamin Blankertz
Gutachter:	Prof. Dr. Klaus-Robert Müller
	Prof. Dr. Thomas Wiegand
	Prof. Dr. Jose C. Principe

Tag der wissenschaftlichen Aussprache: 19. Dezember 2018

Berlin, 2019

To my wife Maika,
for outstanding patience and support.

Es irrt der Mensch, solang' er strebt.
– Johann Wolfgang von Goethe,
Faust I, Vers 317 / Der Herr

Acknowledgements

First and foremost I want to express my gratitude to Prof. Dr. Klaus-Robert Müller for his invaluable guidance, encouragement and support as my advisor over the recent years. I am very thankful to have been granted the opportunity to work in the still young but vivid field of interpretable machine learning, in a time where learning methods seem to become so potent that a wide spread application in daily life is within one's reach, and thus knowledge about the machines' inner reasoning is more sought after than ever before. Klaus' visions, ideas and kind enthusiasm have been a steady source of motivation to me and helped overcoming some of the more tiring periods in my life as a PhD student.

A special thank you goes to Dr. Wojciech Samek, leader of the machine learning group at Fraunhofer HHI, office mate, friend and closest collaborator during my PhD, for always having time for questions, discussions, for providing insights and constructive criticism, and encouraging me to learn and love the inkscape way of figure crafting.

I would also like to thank Prof. Dr. Alexander Binder, without whom this thesis never would have happened. Alexander provided the core intuition for Layer-wise Relevance Propagation and his recommendation has led to my Master's Thesis in Klaus' group at TU Berlin, enabling the pursuit of my PhD in a very favourable environment at TU Berlin and Fraunhofer HHI. I want to thank Alexander for his supervision, his valued scientific input and for honing my resistance to frustration ;)

My gratitude also goes to Dr. Grégoire Montavon, whose thoroughness and research ethics usually meant some extra work, but always paid off in the end in quality and interesting results. Thank you for your support.

Sincere thanks also goes to Dr. Kristof Schütt for setting and always maintaining high standards during our years of study at TU Berlin, which stimulated my competetive spirit to learn and grow. Meeting Kristof undoubtedly has played a big role towards the pursuit of this PhD.

Deep gratitude also goes out to Prof. Dr. Thomas Wiegand – executive director at HHI – and Dr. Thomas Schierl – head of the Video Coding & Analytics Department at HHI – for providing and maintaining an excellent, undisturbed and almost unconstrained research environment.

I would also like to thank Gabriela Thiele and Dominique Jojade for taking care of all my administrative inquiries, sorting out flight connections and always picking the coziest conference accomodations (*i.e.* generally making life easy at HHI).

I thank my (former) colleagues at HHI and TU Berlin for discussions opening up additional perspectives and an enduring pleasurable atmosphere, in no particular order: David Neumann, Leila Arras, Armin Thomas, Miriam Hägele, Vignesh Srinivasan, Sebastian Bosse, Max Kohlbrenner, Sören Becker, Max Alber, Daniel Bartz, Irene Winkler, Simon Wiedemann, Ahmed Magdi Osman, Luis Oala, and others.

I would like to thank Mick Gordon, Chris Hite and Chad Mossholder for the soundtrack to 2016's DOOM, which helped me pick up work on those particularly slow monday mornings.

Finally, I would like to express heartfelt thanks to my wife Maika – for the balance and love she provides, for knowing my moods and needs before even I do, her empathy, our home full of mutts, and for always being right – and to my son Arwed, for the opportunity to play with wooden train sets again without being looked at funnily, and for the experience of growing up a second time. Overall, I consider myself lucky for the long running series of fortunate events bringing me to where I stand today. Given the choice, I would not change a thing.

Abstract

Machine learning techniques such as (Deep) Neural Networks are successfully solving a plethora of tasks, *e.g.* in image recognition and text analysis, and provide novel predictive models for complex physical, biological and chemical systems. However, due to the nested complex and non-linear structure of many machine learning models, this comes with the disadvantage of them acting as a black box, providing little or no information about the internal reasoning. This black box character hampers acceptance and application of non-linear methods in many application domains, where understanding individual model predictions and thus trust in the model's decisions are critically important. In this thesis, we describe a novel method for explaining non-linear classifier decisions by decomposing the prediction function, called Layer-wise Relevance Propagation (LRP). We apply our method to Neural Networks, kernelized Support Vector Machines (with non-linear kernels) and Bag of Words feature extraction pipelines and evaluate LRP theoretically, qualitatively and quantitatively in comparison to other recent methods for interpreting model predictions. Using our method as a tool for comparative analyses between various pre-trained models we reveal different learned prediction strategies and flaws in datasets, predictors and the training thereof.

Zusammenfassung

Techniken des maschinellen Lernens wie (Tiefe) Neuronale Netze lösen eine Vielzahl an Aufgaben mit großem Erfolg, beispielsweise in der Bilderkennung und Textanalyse, und bieten neuartige Vorhersagemodelle für komplexe physikalische, biologische und chemische Zusammenhänge auf. Dies geht jedoch durch die verschachtelte und komplex-nichtlineare Struktur vieler Modelle des maschinellen Lernens mit dem Nachteil einher, dass diese Modelle sich wie Black Boxes verhalten und keine oder nur wenig Informationen über interne Schlussfolgerungen preisgeben. Dieser Black Box-Charakter beeinträchtigt die Anwendung und Akzeptanz von nichtlinearen Methoden in zahlreichen Anwendungsgebieten, in denen das Verstehen individueller Modellvorhersagen, und somit das Vertrauen in das Vorhersagemodell unumgänglich ist. Diese Dissertation behandelt eine neuartige Methode, genannt Layer-wise Relevance Propagation (LRP), zur Erklärung nichtlinearer Klassifikationsentscheidungen mittels der Zerlegung der Vorhersagefunktion. Wir wenden unsere Methode auf Neuronale Netze, Support Vector Maschinen (mit nichtlinearen Kernen) und Bag of Words Merkmalsextraktionssysteme an, und evaluieren LRP auf theoretischer, qualitativer und quantitativer Ebene im Vergleich zu weiteren aktuellen Methoden zur Interpretation von Modellvorhersagen. Unsere Methode als Analysewerkzeug nutzend decken wir vergleichend zwischen diversen vortrainierten Modellen verschiedene erlernte Vorhersagestrategien und Schwächen in Datensätzen, Prädiktionsmodellen und deren Training auf.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Over eight decades ago the first data driven pattern recognition algorithm (Fisher, 1936) was formulated to compute Bayes-optimal decision functions between sample distributions. Early Neural Network type models followed shortly thereafter (McCulloch et al., 1943), mimicking the signaling of organic neurons. Since then, the complexitiy and capabilities of machine learning algorithms have progressed steadily, together with the availability of computational resources and the intricacy and size of problem defining datasets. While in the late 1990s, kernel methods and Support Vector Machines (Vapnik, 1995; Cortes et al., 1995; Schölkopf et al., 1998a) dominated the machine learning (ML) world due to the computationally efficient yet expressive kernel trick (Boser et al., 1992; Müller et al., 1997; Schölkopf et al., 1998b) and the guarantee for sparse yet optimal problem solutions, the popularity of (Deep) Neural Networks was on the rise again in the late 2000s. This reemergence of Neural Network methods is grounded in the availability of very large datasets, their ability to process these excessive amounts of data via the computational capabilities of modern Graphics Processors (GPU); a combination leading to breathtaking performance, especially in the rich and diverse field of Computer Vision (Krizhevsky et al., 2012; Russakovsky et al., 2015). Since then, improvements in network architecture and model capabilities have been steady and fast-paced (Zeiler et al., 2014; Simonyan et al., 2014; Szegedy et al., 2015; Szegedy et al., 2017) and Neural Networks have revolutionized learning-based approaches in other research directions as well, *e.g.* by learning to read subway plans (Graves et al., 2016), understanding quantum many-body systems (Montavon et al., 2013b; Schütt et al., 2017; Schütt et al., 2018), decoding human movement from EEG signals (Sturm et al., 2016; Schirrmeister et al., 2017) and matching or even exceeding human performance in playing games such as Go (Silver et al., 2016), Texas hold'em poker (Moravčík et al., 2017), various Atari 2600 games (Mnih et al., 2015) or Super Smash Bros. (Firoiu et al., 2017).

It is not surprising, that in the near past, high-performance methods quickly found their way out of research labs and are attracting much attention in the industry and media. Methods based on ML and Artificial Intelligence (AI) affect our every day lives, *e.g.* from filling supporting roles in consumer electronics, as simple real time face tracking algorithms in cell phone cameras, to providing high speed tools for stock market prediction impacting global finance markets, and executing supporting and augmentative tasks in medical applications. However in some domains Deep Neural Networks, (kernelized) Support Vector Machines and other complex, non-linear learning algorithms are in general applied and accepted as *black boxes*, providing little information about which aspect of an input sample causes the resulting prediction exactly and in detail. In other fields, this intransparency only allows for cautious application of non-linear models, or none at all due to the associated risk (Caruana et al., 2015; Y. Yang et al., 2018) of intransparent predictions. There is an often retold anecdote, which outlines the tendency of machine learning algorithms to adapt to unexpected or even unintended features within a dataset while apparently solving the posed problem admirably. The earliest known mentioning of this story occurs in (Dreyfus et al., 1992):

> *" [...] In the early days of the perceptron the army decided to train an artificial neural network to recognize tanks partly hidden behind trees in the woods. They took a number of pictures of a woods without tanks, and then pictures of the same woods with tanks clearly sticking out from behind trees. They then trained a net to discriminate the two classes of pictures. The results were impressive, and the army was even more impressed when it turned out that the net could generalize its knowledge to pictures from each set that had not been used in training the net. Just to make sure that the net had*

> *indeed learned to recognize partially hidden tanks, however, the researchers took some more pictures in the same woods and showed them to the trained net. They were shocked and depressed to find that with the new pictures the net totally failed to discriminate between pictures of trees with partially concealed tanks behind them and just plain trees. The mystery was finally solved when someone noticed that the training pictures of the woods without tanks were taken on a cloudy day, whereas those with tanks were taken on a sunny day. The net had learned to recognize and generalize the difference between a woods with and without shadows! Obviously, not what stood out for the researchers as the important difference."*

No further references are given in (Dreyfus et al., 1992) for above tale. As far as we know, this anecdote might even be entirely fabricated. However, it demonstrates well some of the major shortcomings of many machine learning algorithms: A model performing well *in number* (*e.g.* accuracy) in a laboratory setting might not decide necessarily based on *meaningful* features of the given data, potentially rendering it useless outside the lab. Furthermore, what *meaningful* means to a trained model does not necessarily have to match the expectation of the researcher who trained the model to solve a given task (in a specific way). We note that amusingly, already (Rosenblatt, 1957) refers to the Perceptron, fitted with sensors for input and a signalling apparatus to communicate its outputs, as a *black box*, even though with a slightly different meaning.

With AI systems further penetrating life in the role of supporting agents for critical tasks, concern regarding the trustworthiness of such systems is being voiced. While systems providing *yes*/*no*-answers might be sufficient for consumer-level electronics, simple binary decisions are often of only limited value in contexts where mispredictions are tied to considerable monetary losses, are endangering the life and health of humans and animals and *trust* in the system is essential, *e.g.* in the medical domain (Caruana et al., 2015) or with autonomously driving vehicles. A lack of this information for the verification of the agent's decision by a human expert via early identification, avoidance and eradication of potential flaws would submit *e.g.* patients to the risk of (life threatening) misdiagnoses or jeopardize the well-being of motorists and pedestrians. In research in general, knowledge about the *what*, *where* and *how* behind the decision of an automated system are of critical importance. The European Union – recognizing the potential benefits from Artificial Intelligence (AI) supported systems as well as the dangers – has extended its 2016 Parliament's General Data Protection Regulation (GDPR). The expanded version has taken effect on May 25, 2018, adding the *right to explanation* to the policy in Articles 13, 14 and 22 (Parliament and Council of the European Union, 2016; Goodman et al., 2017), highlighting the importance of human-understandable interpretations derived from machine decisions additionally. With models from ML known to exceed the capabilities of humans performing the same tasks such as in image categorization (Karpathy, 2014; Ioffe et al., 2015; Szegedy et al., 2016; He et al., 2016; Szegedy et al., 2017) or playing games (Silver et al., 2016; Moravčík et al., 2017; Mnih et al., 2015; Firoiu et al., 2017), gaining insight into the models' reasoning could allow humans to learn from the machine and discover unintuitive and novel ways of reasoning.

In recent years – especially after the explosive revival of the Deep Neural Networks – attention towards the problem of intransparency in machine learning models has greatly increased. However, fully understanding complex models still is an unsolved problem.

One goal of this thesis is the introduction of the novel and general concept of *Layer-wise Relevance Propagation* (LRP), a method for the derivation of *explanations* for individual (machine learning) model predictions. The method operates by *decomposing* the output of the evaluated prediction function, allowing for conclusions about the reason behind a particular model decision wrt to the given input sample. For that, we define a quantity called *relevance*, which is motivated by the component-wise contributions of an input sample and transparently obtainable for the prediction of linear models. Following that principle, LRP explains the model decision by attributing parts of the decomposed predictor output to components of the input responsible for the prediction. The result is a human-interpretable *relevance map* for all elements of the model input, illustrating their differential contributions to the inference output. Figure 1.1 illustrates the general process of an application of LRP to a model decision, and the information obtained from a relevance map, at hand of an image recognition task.

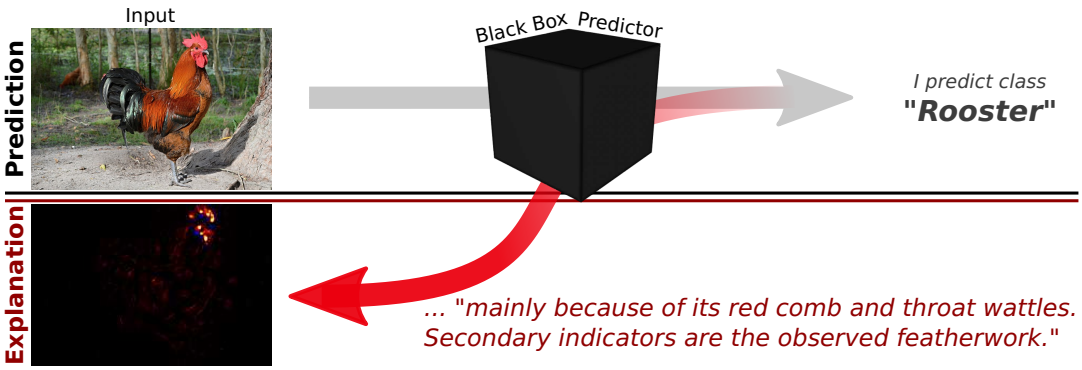

FIGURE 1.1: **An illustration of the general process of applying LRP.** A (black box) model makes a prediction based on a given input, as shown here in an image recognition setting. LRP then decomposes the model's evaluated prediction function into differential contributions for all input components, by performing a backward pass through the model. The resulting relevance map can then optionally be visualized for interpretation by a human observer.

We discuss desirable properties for methods of model interpretability and compare LRP theoretically and empirically to other methods. Applying the method to different families of prediction functions, we use it as a tool for analyzing the reasoning of contemporary state of the art models from image recognition. Here, we gain insight into different learned prediction strategies and how they depend on choices made for data preprocessing and model initialization, and flaws in the training and test data. Finally, LRP is embedded into the novel framework of *Spectral Relevance Analysis* (SpRAy), allowing for time-efficient, systematic and semi-automatic analyses of relevance maps computed over large sets of data.

1.1 Structure of this Thesis

Over the course of this work, we describe the Layer-wise Relevance Propagation method for explaining non-linear classifiers, investigate its properties theoretically and empirically, and analyze the reasoning of several image recognition classifiers insightfully via the method's application. The chapters of this thesis are organized as follows:

Chapter 2 (Decomposing Non-Linear Classifiers with Layer-wise Relevance Propagation) first motivates and describes the *Layer-wise Relevance Propagation* (LRP) method for decomposing classifier decisions as a general concept. Then, the derived basic decomposition rules are applied to the building blocks of Deep Neural Networks and Bag of Words pipelines with SVM predictors. For both model types, example applications set in the the domain of image recognition demonstrate the meaningfulness of relevance maps computed with LRP.

Chapter 3 (Evaluating and Understanding Heatmaps) raises the question of what defines a good explanatory heatmap. The chapter introduces a method for measuring heatmap quality based on input perturbations and compares several interpretability methods theoretically, qualitatively and quantitatively.

Chapter 4 (Comparing Fisher Vector SVMs and Deep Neural Networks) extends LRP decomposition rules for Bag of Words models to the popular Fisher vector feature mapping and comparatively analyzes prediction behaviour of a Fisher vector model for image recognition to several Deep Neural Network architectures, among other things in terms of dependency on image context.

Chapter 5 (Investigating DNNs for Face Categorization) summarizes recent developments in age and gender recognition from face images using Deep Learning methods. Establishing

optimized decomposition rules and rule parameterizations for decomposing the decisions of sequential Deep Neural Networks, this chapter investigates how a key choices made for model training affect the performance and prediction strategy of Neural Network classifiers.

Chapter 6 (Ensemble Relevance Map Analysis of Classifier Behaviour) presents a novel method – *Spectral Relevance Analysis* (SpRAy) – combining the relevance maps from LRP as features with Spectral Clustering, enabling efficient dataset scale analyses of models' prediction behaviours. SpRAy is used to further investigate the classifiers from Chapter 4 on Pascal VOC data, identifying previously unknown artefacts in the models' prediction strategies.

Chapter 7 (Conclusion) concludes this thesis with a summary and provides a brief overview over the application of LRP in the sciences and an outlook to future work.

1.2 Own Contributions

Conceptual and Technical Contributions

1. The principle of Layer-wise Relevance Propagation (LRP) is proposed (S. Bach et al., 2015), generalizing the nature of interactive interpretability of linear methods to non-linear stacks of feature mappings and transformations.

2. A proposition of approximate function decomposition based on Taylor expansion (S. Bach et al., 2015), as a first step towards the Deep Taylor Decomposition framework.

3. Subsequent applications of the LRP principle to the layers of Deep Neural Networks and Bag of Words feature mappings – in particular the improved Fisher vector coding – and (kernelized) Support Vector classifiers are derived (S. Bach et al., 2015; Lapuschkin et al., 2016a; Binder et al., 2016a; Binder et al., 2016b).

4. The derivation of advanced and purposed decomposition rule variants of the basic principle of conservative relevance decomposition are proposed and decomposition rule parameterizations are recommended (S. Bach et al., 2015; Lapuschkin et al., 2016a; Binder et al., 2016a; Lapuschkin et al., 2017; Samek et al., 2017).

5. Software toolboxes with implementations of the LRP algorithm for Matlab, Python, the Caffe deep learning framework and Keras are provided (Lapuschkin et al., 2016b; Alber et al., 2018a; Alber et al., 2018b).

6. Spectral Relevance Analysis (SpRAy) is proposed as a semi-automatic method for efficient analysis of patterns of model behaviour on large bodies of data as a novel technical contribution (Lapuschkin et al., 2018).

Theoretical Contributions

1. Desiderata of interpretability methods are discussed. Methods for interpretability are contrasted wrt to the discussed properties (S. Bach et al., 2015; Samek et al., 2017).

2. A quantitative measure for the quality of explanatory heatmaps is proposed alongside a general perturbation-based evaluation technique (S. Bach et al., 2015; Lapuschkin et al., 2016a; Samek et al., 2017).

Experimental Contributions

1. A qualitative and quantitative comparison of heatmaps from different models and interpretability methods (Lapuschkin et al., 2016a; Samek et al., 2017).

2. Analyses of the learned prediction strategies between a state of a art Fisher vector Bag of Words predictor and a DNN on the Pascal VOC dataset, using explanatory relevance maps (Lapuschkin et al., 2016a).

3. Comparisons of predictive behaviour of various DNN models by architecture on generic photographic images from the ImageNet (Lapuschkin et al., 2016a) and face images for age and gender recognition from the Adience benchmark dataset (Lapuschkin et al., 2017), using visualized relevance maps.

4. Assessment of the impact of pretraining and input preprocessing on various DNN architectures, via performance measures and relevance maps (Lapuschkin et al., 2017).

5. Manual and automatic large-scale analyses of model behaviour for several DNNs and the FV model on ImageNet and Pascal VOC, *e.g.* leading to the discovery and verification of learned prediction artefacts and flaws in the data and processing pipelines (Lapuschkin et al., 2016a; Lapuschkin et al., 2018)).

Contributions not included in this Thesis

1. The (theoretical) advancements of (simple) Taylor Decomposition into the framework of *Deep Taylor Decomposition* (Montavon et al., 2016; Montavon et al., 2017).

2. Investigations of human stride patterns for subject identification on a range of (non-linear) predictors to verify the meaningfulness of learned biometrical features (Horst et al., 2018).

3. The generation of a novel spoken digit dataset and subsequent comparative analysis of DNNs trained on raw waveform and spectrogram features (Becker et al., 2018).

4. Analysis of model reasoning for human action recognition in compressed domain using Fisher vector models (Srinivasan et al., 2017).

5. A first application of LRP to raw EEG signal analysis with DNN has shown that the network models learned physiologically meaningful features. In contrast to the baseline method CSP-LDA, which only provides explanation pattern for each subject (class), LRP is able to compute explanations on a spatio-temporal resolution per test trail (Sturm et al., 2016).

6. An application of LRP to one class support vector predictors for localizing structurally anomalous cloud application behaviour in communication graphs (Schwenk et al., 2014).

1.3 List of Publications

The following contains a list of contributions made by the author to the field of explainable artificial intelligence in machine learning. As it is common practice in this field, some of the materials presented within this thesis have been published in journals or presented at peer-reviewed conferences. This thesis will focus mainly on the work presented in (Bach et al., 2015), (Lapuschkin et al., 2016a), (Bach et al., 2016), (Samek et al., 2017), (Lapuschkin et al., 2017) and (Lapuschkin et al., 2018) while borrowing results, figures and ideas from other, in part unpublished works of the author. Note that in 2016, the last name of this thesis' author has changed from *Bach* to *Lapuschkin*, which also reflects in the publications listed below and the contributions listed above. I would like to thank my co-authors for allowing me to use parts of text from previous publications.

Journal Articles

Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek (2015). "On Pixel-wise Explanations for Non-Linear Classifier Decisions by Layer-wise Relevance Propagation". In: *PLoS ONE* 10.7, e0130140.

Horst, F., S. Lapuschkin, W. Samek, K.-R. Müller, and W. I. Schöllhorn (2018). "Explaining the Unique Nature of Individual Gait Patterns With Deep Learning". In: *Scientific Reports*. In Revision.

Lapuschkin, S., A. Binder, G. Montavon, K.-R. Müller, and W. Samek (2016b). "The Layer-wise Relevance Propagation Toolbox for Artificial Neural Networks". In: *Journal of Machine Learning Research* 17.114, pp. 1–5.

Lapuschkin, S., S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller (2018). "Assessing What Machines Really Learn – Unmasking 'Clever Hans' Predictors". In: *Nature Communications*. In Revision.

Montavon, G., S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller (2017). "Explaining NonLinear Classification Decisions with Deep Taylor Decomposition". In: *Pattern Recognition* 65, pp. 211–222.

Samek, W., A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller (2017). "Evaluating the visualization of what a Deep Neural Network has learned". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.11, pp. 2660–2673.

Sturm, I., S. Lapuschkin, W. Samek, and K.-R. Müller (2016). "Interpretable Deep Neural Networks for Single-Trial EEG Classification". In: *Journal of Neuroscience Methods* 274, pp. 141–145.

Peer-reviewed Contributions to Conferences

Alber, M., S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, and P.-J. Kindermans (2018a). "How to iNNvestigate Neural Networks' predictions!" In: *Machine Learning Open Source Software (MLOSS): Sustainable Communities, NIPS Workshop*.

Bach, S., A. Binder, K.-R. Müller, and W. Samek (2016). "Controlling Explanatory Heatmap Resolution and Semantics via Decomposition Depth". In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 2271–2275.

Binder, A., W. Samek, G. Montavon, S. Bach, and K.-R. Müller (2016c). "Analyzing and Validating Neural Networks Predictions". In: *Proceedings of the ICML'16 Workshop on Visualization for Deep Learning*, pp. 1–4.

Lapuschkin, S., A. Binder, G. Montavon, K.-R. Müller, and W. Samek (2016a). "Analyzing Classifiers: Fisher Vectors and Deep Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2912–2920.

Lapuschkin, S., A. Binder, K.-R. Müller, and W. Samek (2017). "Understanding and Comparing Deep Neural Networks for Age and Gender Classification". In: *Proceedings of the ICCV'17 Workshop on Analysis and Modeling of Faces and Gestures (AMFG)*.

Montavon, G., S. Bach, A. Binder, W. Samek, and K.-R. Müller (2016). "Deep Taylor Decomposition of Neural Networks". In: *Proceedings of the ICML'16 Workshop on Visualization for Deep Learning*, pp. 1–3.

Samek, W., G. Montavon, A. Binder, S. Lapuschkin, and K.-R. Müller (2016). "Interpreting the Predictions of Complex ML Models by Layer-wise Relevance Propagation". In: *Proceedings of the Interpretable ML for Complex Systems NIPS'16 Workshop*, pp. 1–5.

Srinivasan, V., S. Lapuschkin, C. Hellge, K.-R. Müller, and W. Samek (2017). "Interpretable Human Action Recognition in Compressed Domain". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1692–1696.

Book Chapters

Binder, A., S. Bach, G. Montavon, K.-R. Müller, and W. Samek (2016a). "Layer-wise Relevance Propagation for Deep Neural Network Architectures". In: *Information Science and Applications (ICISA) 2016*. Ed. by K. J. Kim and N. Joukov. Vol. 376. Lecture Notes in Electrical Engineering. Singapore: Springer Singapore, pp. 913–922. ISBN: 978-981-10-0557-2.

Binder, A., G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek (2016b). "Layer-wise Relevance Propagation for Neural Networks with Local Renormalization Layers". In: vol. 9887. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 63–71.

Preprints

Alber, M., S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, and P.-J. Kindermans (2018b). "iNNvestigate Neural Networks!" In: *CoRR* abs/1808.04260. arXiv: 1808.04260.

Becker, S., M. Ackermann, S. Lapuschkin, K.-R. Müller, and W. Samek (2018). "Interpreting and Explaining Deep Neural Networks for Classification of Audio Signals". In: *CoRR* abs/1807.03418. arXiv: 1807.03418.

Schwenk, G. and S. Bach (2014). "Detecting Behavioral and Structural Anomalies in Media-Cloud Applications". In: *CoRR* abs/1409.8035. arXiv: 1409.8035.

Chapter 2

Decomposing Non-Linear Classification Functions with Layer-wise Relevance Propagation

Many problems in machine learning benefit greatly from using non-linear methods and preprocessing steps over just linear methods, which often are used as a first baseline for comparison (*e.g.* (LeCun, 1998)). Especially Deep Learning Models have significantly grown in complexity and capabilities in the recent past (Krizhevsky et al., 2012; Mnih et al., 2015; Szegedy et al., 2017; Schütt et al., 2017; Firoiu et al., 2017), enabling them to solve more difficult tasks in the first place. However, (Deep) Neural Networks and non-linear predictors in general are – as so-called *black box models* – only restrictedly fit for use in many industrial or research fields. This is true when an interpretation of the model's prediction is as important – or even more important – than the prediction itself, or a performance estimate over a benchmark data set. In case of a classification task, a simple *yes*-or-*no* answer is sometimes of limited value in applications where questions about *which* features are picked up by the predictor, *where* (or *when*) are those features and *how* is it structured, must be answerable in order for *e.g.* a human expert to verify the prediction or to apply further processing on that information.

It seems to be commonly accepted that a trade-off exists between the (potential) capabilities of a model and model transparency or interpretability (Lou et al., 2012; Burrell, 2016; Goodman et al., 2017; Lipton, 2018). Even recent work in *e.g.* computer security (Schütt et al., 2012; Arp et al., 2014), brain computer interfacing (BCI) (Parra et al., 2005; Blankertz et al., 2008; Blankertz et al., 2011; Haufe et al., 2014), human gait analysis (Phinyomark et al., 2017; Slijepcevic et al., 2018), biology and healthcare (Y. Yang et al., 2018) rely on linear methods due to their explainability. (Horst et al., 2017) and (Phinyomark et al., 2017) conclude that (for that reason), "for [human gait] classification, the most popular supervised learning method is the [linear] SVM classifier" (Phinyomark et al., 2017), citing numerous works from within the decade. In above examples, the choice of model is governed and limited mainly by the required transparency of the predicting function and the so-called *semantic gap* (Sommer et al., 2010) between the prediction results and their interpretatability by a (human) expert.

In this chapter we
(1) introduce a quantity R called *relevance* which is motivated by the interpretability of input component *importance* (as its *contribution* to the function output) which is readily available for linear predictors (Section 2.1).
(2) We further propose and describe *Layer-wise Relevance Propagation* (LRP) as a general and novel concept for analyzing the predictions for individual input samples using graph-shaped model architectures by *decomposing* the model's prediction function $f(\mathbf{x})$ as components of a sum, motivated by the interpretability of linear models (Section 2.2).
(3) (Simple) *Taylor Decomposition* is introduced as an approximate means for decomposing functions (or parts thereof) which do not fit the structure assumed by LRP, constituting a first step towards the *Deep Taylor Decomposition* (Montavon et al., 2016; Montavon et al., 2017) framework.
(4) Finally, we show that LRP yields meaningful results when applied to a wide range of non-linear model types, such as (Deep) Neural Networks (Section 2.3) and
(5) Bag of Words pipelines with SVM predictors (Section 2.4).

The application focus of this thesis lies on tasks from the computer vision domain. We thus make use of the corresponding terminology, *e.g.* refering to the inputs of a model as pixels, although the application of LRP is not limited to the domain of computer vision or image recognition. For simplicity, we also chose to adapt to the terminology of Neural Networks[1] when describing LRP generally in Section 2.2.

This chapter covers the contributions from (S. Bach et al., 2015; Samek et al., 2016; Binder et al., 2016a), as implemented in (Lapuschkin et al., 2016b) and (Alber et al., 2018a; Alber et al., 2018b).

2.1 Interpreting Linear Predictors

Linear predictors are still a popular choice for solving machine learning problems, especially in applied computer security (Sommer et al., 2010; Schütt et al., 2012; Arp et al., 2014) or biology and healthcare (Y. Yang et al., 2018), despite Deep Learning techniques greatly increasing the capabilities of data driven decision making. Reasons for that popularity are the simplicity of linear models, the comparatively low computational effort required for training and prediction and the straightforwardness and interpretability of the learned decision making rules and how they affect each individual data point during inference.

Suppose a linear classifier which has been trained on some example data to separate samples belonging to two classes, with the prediction being thresholded at 0. The model consists of a learned weight vector $\boldsymbol{w} \in \mathbb{R}^D$ and a scalar bias term b. The vector $\boldsymbol{w}$ is a normal vector describing a hyperplane in data space, separating both sample classes from one another. The value b describes the distance of the shift between the hyperplane and the coordinate origin in data space. Such a linear model operating in $\mathbb{R}^2$ can be seen in Figure 2.1 (left).

Given an input sample $\boldsymbol{x} \in \mathbb{R}^D$ the model can make a prediction $f(\boldsymbol{x})$ by computing the projection of $\boldsymbol{x}$ onto $\boldsymbol{w}$ as a dot product and adding the constant shift b.

$$f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b = y \tag{2.1}$$

In a two-class classification setting, usually the sign of the output y determines the estimated class membership of the given $\boldsymbol{x}$. Since the function only depends on a dot product between the input sample $\boldsymbol{x}$ and the learned parameters $\boldsymbol{w}$ – *i.e.* a sum over component-wise multiplications $w_d x_d$ for all d – and a constant addend b, we can write the prediction for a given data point as

$$f(\boldsymbol{x}) = \sum_d w_d x_d + b \,. \tag{2.2}$$

We immediately see how each component x_d of $\boldsymbol{x}$ contributes *in interaction* with the corresponding parameters w_d to the output of the prediction function, and what influence the constant bias term b has, *i.e.* how *relevant* each input component x_d is to the evaluation of $f(\boldsymbol{x})$ as $w_d x_d$, and in what way. The learned parameters $\{\boldsymbol{w}, b\}$ on their own inform how sensitive the model is to changes in the input space, which is used in methods of Sensitivity Analysis (Baehrens et al., 2010; Simonyan et al., 2013).

Let us define the quantity of *relevance* R as a measure of how (much) an input unit contributes to the prediction of a model, as an addend to a sum of (relevance) contributions. For example, in context of above linear model $R_d = w_d x_d$, $R_b = b$ and

$$f(\boldsymbol{x}) = \sum_d R_d + R_b \,. \tag{2.3}$$

A positive value $R_d > 0$ then means that x_d of the given $\boldsymbol{x}$ positively contributes to $f(\boldsymbol{x})$, *i.e.* wrt to the learned model, x_d speaks for the prediction of the positive class. Similarly, a $R_d < 0$ signifies that x_d contradicts the prediction of the positive class, or, encourages the

[1] *E.g.* we use the terms "neuron" and "layer" as general proxies for input components or more intermediate representations of the data or corresponding model elements.

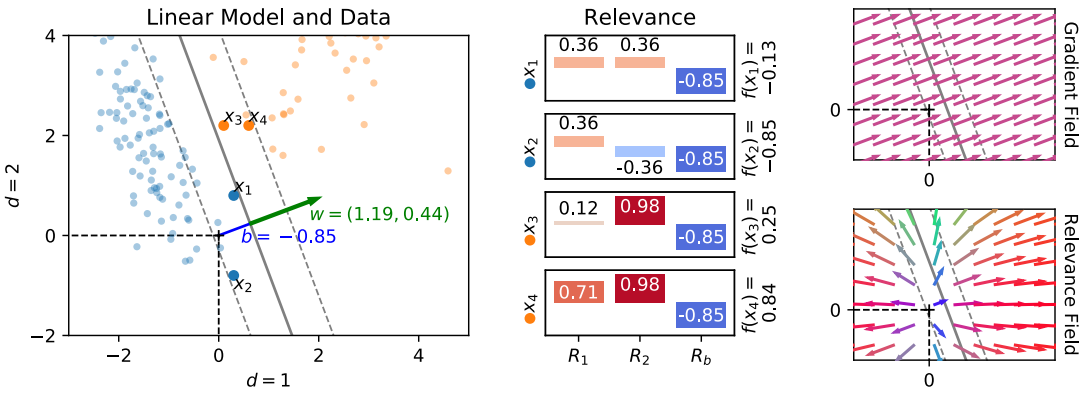

FIGURE 2.1: **Linear models are inherently interpretable.** *Left:* A linear model $\{\boldsymbol{w}, b\}$ has been trained to separate two point clouds. Training data points are translucent and labelled via colour coding. The test samples $\{x_1 = (0.3, 0.8), x_2 = (0.3, -0.8), x_3 = (0.1, 2.2), x_4 = (0.6, 2.2)\}$ shown as larger, opaquely coloured dots are predicted by projection against $\boldsymbol{w}$ and thresholding against b. *Center:* A decomposition of the prediction function f into relevance values for all test samples. The constant bias term b correspondingly is attributed identical relevance values R_b for each test point prediction. Relevance attributions differ considerably between test sampels, despite the (linearly projecting) prediction function being the same for all points, due to changes in the composition in the prediction outcome. *Right:* Vector fields visualizing the (gradient of) function f (*Gradient Field*) and relevance attributions (*Relevance Field*) on a grid of locations from within the window shown in the leftmost graphic. Arrows show the $\mathbb{R}^2$ data space components of the unit vectors in $\mathbb{R}^3$ for both gradients and relevance values. Arrows are color-coded as mixtures of rgb-values wrt to the relative imporance of $d = 1$ (red channel), $d = 2$ (green channel) and the bias b (blue channel). For individual test samples, relevance values are informative about how each component affects the outcome of $f(\mathbf{x})$ (even *e.g.* whether the bias dominates the prediction: Blue arrows), while the function's gradient is not informative in that respect and identical at any location in input space.

prediction of the negative class. A value $R_d \approx 0$ identifies features x_d (almost) not affecting the prediction outcome[2]. Figure 2.1 (center) shows relevance values computed for the test samples in Figure 2.1 (left).

Relevance is not a binary measure. So can for example a $R_{d_1} \gg R_{d_2} > 0$ tell us that x_{d_1} and x_{d_2} both describe features speaking for a prediction of $f(\mathbf{x}) > 0$, but x_{d_1} more distinctively affects the evaluation of f. For the computation of relevance values, the constant bias value b is treated equivalently to the data-dependant inputs x_d resulting in the attribution of R_b to b.

In contrast to the gradient of the learned model (*e.g.* in Sensitivity Analysis) which yields the same results for all possible locations in dataspace for above linear model, the decomposition of $f(\mathbf{x})$ into relevance values is individual to each model input. Figure 2.1 (right) provides a contastive graphical interpretation of relevance as a function of f and $\mathbf{x}$ and the gradient of the model (Sensitivity Analysis). Relevance is a more informative measure for assessing the *composition* about the prediction of individual data points than looking at (the gradient of) the function, providing insight to the sensitivity of the model to given inputs.

2.2 Generalizing Relevance Decomposition for Non-linear and Multi-layer Predictors

The goal of component-wise relevance decomposition is to understand the contribution of a single component of an input $\mathbf{x}$ to the prediction $f(\mathbf{x})$ made by a classifier f in a classification (or regression) task. We would like to find out, separately for each sample $\mathbf{x}$, which input

[2] In even simpler words: x_d increases $f(\mathbf{x})$ if $R_d > 0$ and decreases it if $R_d < 0$, et cetera.

components or dimensions contribute to what positive or negative extend to a classification result by quantitative measure. We assume that the classifier has real-valued outputs which are thresholded at zero. In such a setup the classifier is a mapping $f : \mathbb{R}^D \rightarrow \mathbb{R}^1$ such that $f(\mathbf{x}) > 0$ (as it is common practice) denotes presence of the learned structure. The important constraint specific to classification consists in finding the differential contribution relative to the state of maximal uncertainty $f(\hat{\mathbf{x}}) \approx 0$. One possible way is to decompose the prediction $f(\mathbf{x})$ as a sum of terms of the separate input dimensions x_d

$$f(x) \approx \sum_{d=1}^{D} R_d \tag{2.4}$$

as quantities R_d we call *relevance*. The qualitative interpretation for relevance scores is that $R_d < 0$ contributes evidence against the presence of a structure which is to be classified while $R_d > 0$ contributes evidence for its presence, as already noted in above Section 2.1.

The trivial decomposability of linear models is the key motivator for the *Layer-wise Relevance Propagation* (LRP). For deeper models, especially with non-linear transformations being involved, such a decomposition over multiple layers of computation is not naturally given. With LRP, we aim to derive a set of rules, such that decompositions as in Equation (2.4) can be computed for non-linear models meaningfully, allowing for an interpretation of individual model decisions likewise to simple linear models, in order to eliminate the trade-off between model performance (via complexity) and interpretability.

The method(s) proposed within the scope of this thesis do not involve or require segmentation or pixel-wise training (in the context of image recognition) as learning setup or pixel-wise labeling for the training phase. Instead, the methods described here do not require specialized model training or do interfere with the training process at all and are able to operate on pretrained models in a semi-automatic, unsupervised manner (*i.e.* labels are not required), as will be demonstrated in following chapters.

This section will focus on the derivation of the LRP framework as a general method for decomposing prediction functions, and the assumptions made for the decomposed predictors. We introduce Taylor Decomposition as an variant of LRP, based on Taylor expansion, which allows for an (approximate) decomposition of functions not fitting the pattern of rules defined within the LRP framework. We conclude the current section with the extension of the LRP framework with a set of purposed variants to the basic decomposition approach.

Thereafter, we will show that LRP yields meaningful results for a wide range of non-linear classification architectures, such as Bag of Words models and Deep Neural Networks in the context of image recognition tasks.

2.2.1 Conservation of Relevance across Layers of Computation

We will describe Layer-wise Relevance Propagation as a concept defined by a set of constraints. We assume that the classifier can be decomposed into several layers of computation. Such layers can be parts of a feature extraction algorithm as part of an image processing pipeline or a classification algorithm run on the precomputed features. The first layer of the model to decompose is the input layer, *e.g.* receiving as inputs pixels from image, documents of text or other data, and the last layer of the model f is the real-valued prediction output. We assume that the intermediate representation of the data at the l-th layer is modeled as vector (or matrix, tensor, or single neuron) $\mathbf{z} = (z_i^{(l)}) \in \mathbb{R}^{\dim(l)}$. LRP assumes that we have a relevance score $R_j^{(l+1)}$ for each component $z_j^{(l+1)}$ of the representation $\mathbf{z}$ at layer $(l+1)$. The objective is to compute relevance scores $R_i^{(l)}$ for each component $z_i^{(l)}$ at layer (l), which is one "step" closer to the input layer, such that the following equation holds.

$$\sum_d R_d^{(\text{input})} = \cdots = \sum_i R_i^{(l)} = \sum_j R_j^{(l+1)} = \cdots = f(\mathbf{x}) \tag{2.5}$$

Iterating Equation (2.5) from the classifier output $f(\mathbf{x})$ to the input layer yields Equation (2.4) and the desired sum decomposition of $f(\mathbf{x})$ into differential contributing parts $R_d^{(1)}$ for each

component of the input, while conserving the relevance sum over all layers. However, as we will show, a decomposition satisfying Equation (2.5) *alone* is neither per se unique, nor guaranteed to provide *meaningful* decompositions. We provide simple counter examples: Again, suppose a linear model as in Section 2.1. Let us define the output layer relevance score as $R_1^{(2)} = f(\mathbf{x})$. One possible relevance decomposition satisfying Equation (2.5) is

$$R_d^{(1)} = \begin{cases} R_1^{(2)} & \text{if } d = 1 \\ 0 & \text{else} \end{cases} \tag{2.6}$$

which clearly does not provide any meaningful analysis since all relevance is pooled into one single dimension at the input layer. The second example

$$R_d^{(1)} = f(\mathbf{x}) \frac{|w_d x_d|}{\sum_{d'} |w_{d'} x_{d'}|} \tag{2.7}$$

also satisfies Equation (2.5), yet all relevances attributions $R_d^{(1)}$ have the same sign as the prediction $f(\mathbf{x})$. Thus, all inputs are to be interpreted as relevant to the predicted class, which is not a realistic interpretation for many classification problems (see Figure 2.1).

2.2.2 Relevance Decomposition as a Function Proportional to Forward Contributions

Let us discuss a more meaningful way of defining relevance decomposition. The linear example from Section 2.1 and its decomposition into relevance values in Equation (2.3) provide a good intuition about the nature of relevance R and what it should be, namely a measure for a *local contribution of a neuron* to a prediction $f(\mathbf{x})$.

LRP assumes that the prediction function f computes mappings z_{ij} between intermediate representations of the input data, from components $z_i^{(l)}$ to immediately succeeding mapping outputs j of $z_j^{(l+1)}$ at layer $(l+1)$, and can thus be expressed as such. All mappings with a common output component j are aggregated at j before being propagated further, *e.g.* by summation

$$z_j = \sum_i z_{ij} \text{ , s.t.: } i \text{ is mapping input to } j \tag{2.8}$$

Each mapping z_{ij} thus *contributes its share* to the aggregation (or activation) z_j at output j. This pattern can be found in almost any model, *e.g.* in between the neurons in neighbourig layers of an Artificial Neural Network as $z_{ij} = x_i w_{ij}$ in linear or convolutional layers, or generic (Bag of Words) feature mapping functions, which will both be discussed in detail later. The intermediate goal of LRP is to compute relevance scores $R_i^{(l)}$ – the local contributions of a component i to its immediate successors j, and transitively to the evaluated function $f(\mathbf{x})$ – for any component i at any layer (l) within the model. Figure 2.2 illustrates part of a network-shaped classifier with multiple layers of nodes with directed connections. To achieve this, we express the relevance $R_i^{(l)}$ attributed to each layer (l) and node i in terms of relevance messages $R_{i\leftarrow j}^{(l,l+1)}$, which correspond to the appropriate forward mappings z_{ij} and depend on the given relevance values $R_j^{(l+1)}$ of the succeeding layer. The relevance messages are, however, directed from a neuron towards its input neurons, in contrast to the direction of the forward mappings z_{ij}.

In order for the total amount of relevance to be conserved locally, we must ensure that relevance is neither lost nor created in the process of decomposition. For that, we characterize the behaviour of LRP by considering single neurons. Firstly, we define a local relevance conservation constraint over the relevance messages being backpropagated from node j at layer $(l+1)$ to its inputs i at layer (l):

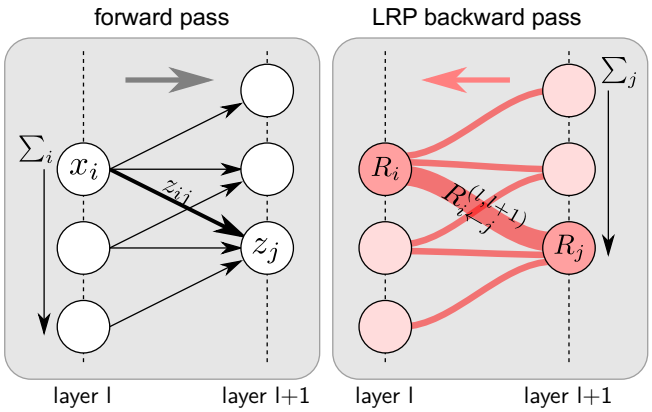

FIGURE 2.2: **LRP considers the model it is applied to as a graph of directed numerical mappings.** Mappings z_{ij} contributing significantly towards the activation of important nodes in the forward pass will in turn carry correspondingly proportioned *relevance messages* $R_{i\leftarrow j}^{(l,l+1)}$ towards input nodes during the application of LRP.

$$\sum_i R_{i\leftarrow j}^{(l,l+1)} = R_j^{(l+1)}\,,\ \text{s.t.: } i \text{ contributes to } j \tag{2.9}$$

Relevance messages satisfying Equation (2.9) can generally be written as

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)} \tag{2.10}$$

where z_{ij}/z_j measures the *proportional contribution* of the forward message z_{ij} to the activation z_j of output component j. Equation (2.10) formulates the basic building block for any decomposition rule and its extensions.

The intuition behind Equation (2.10) is as follows: If a mapping z_{ij} contributes (strongly) to the overall trend z_j of the decomposed mapping (*i.e.* $\text{sign}(z_{ij}) = \text{sign}(z_j)$), the relevance message $R_{i\leftarrow j}^{(l,l+1)}$ should receive a (large) and positively weighted part of the relevance from $R_j^{(l+1)}$. Should z_{ij} contradict the general mapping trend, $R_{i\leftarrow j}^{(l,l+1)}$ would receive a negatively weighted part of the top layer relevance correspondingly. In both cases of $z_{ij} = 0$ and $R_j^{(l+1)} = 0$, the relevance backward message will also be zero, since either i does not contribute to j at all in the former or j has no significance to the model output in the latter case.

The exact propagation rule or relevance message definition depends on the type, position and function of a component within architecture of the computational graph. Applications to DNN and Bag of Words mapping functions will be derived in Sections 2.3 and 2.4 respectively. Purposed variations of Equation (2.10) will be discussed in Section 2.2.6. Generally, the backwards-directed relevance messages $R_{i\leftarrow j}^{(l,l+1)}$ should fulfill the following structure for local relevance conservation to hold:

$$R_{i\leftarrow j}^{(l,l+1)} = v_{ij} R_j^{(l+1)} \text{ with } \sum_i v_{ij} = 1 \tag{2.11}$$

After the decomposition step has been established for LRP, we define the relevance of a node i at layer (l) as the sum of incoming relevance messages from direct successor components at layer $(l+1)$, *i.e.* an aggregation of all its contribution relevances to successor neurons:

$$R_i^{(l)} = \sum_j R_{i\leftarrow j}^{(l,l+1)}\,,\ \text{s.t.: } i \text{ contributes to } j \tag{2.12}$$

The output nodes $R_k^{(L)}$ at the last layer (L) are an exception to this rule, as they can not be the input to any other component. Further, since it is the objective to *decompose* the value $f(\boldsymbol{x})$ of the output node, we initialize the method with $R_k^{(L)} = f(\boldsymbol{x})$ (or rather $f(\boldsymbol{x})_k$ in case multiple model outputs exist and a decomposition only for output k is desired).

In combination, Equations (2.9) to (2.12) describe relevance score assignment from a layer $(l+1)$ to its direct predecessor layer (l).

$$R_i^{(l)} = \sum_j R_{i\leftarrow j}^{(l,l+1)} = \sum_j \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)} \tag{2.13}$$

That is, a node is deemed relevant if it (transitively) contributes to neurons that are relevant themselves. Thus, LRP aims to analyze the *interaction between model and input data*. This procedure is applied iteratively, layer-by-layer, from a model output of choice, until the input of the computational graph is reached. The application of the basic rules of LRP to a layered, sequential model can be expressed as a simple algorithm (see Algorithm 1):

Algorithm 1: Layer-wise Relevance Propagation

Data: $R^{(L)} = f(\boldsymbol{x})$
model parameters and/or mappings z_{ij} , z_j for all layers
Result: $\forall i, l : R_i^{(l)}$
for $l \in \{L-1, \ldots, 1\}$ **do**
  $\forall i, j : R_{i\leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)}$;
  $\forall i \quad : R_i^{(l)} = \sum_j R_{i\leftarrow j}^{(l,l+1)}$;
end

For single layers (l), the decompositions performed with LRP are step-wise linear (see Equations (2.11) and (2.13). Due to the potentially non-linearly activated inputs $\boldsymbol{x}$ (as outputs of preceding layers) as well as backpropagated relevance quantities $R_j^{(l+1)}$ from upper layers, computed wrt to the potentially non-linear output activations of (l) itself, however, LRP becomes a non-linear decomposition method. Derivations can be found in Section 2.3.2 and example applications to non-linear models such as Deep Neural Networks and Bag of Words methods are presented in Sections 2.3.3 and 2.4.3 respectively.

2.2.3 Verifying: LRP for Linear Models

Above description of the decompositions computed by LRP rely on arbitrary mappings z_{ij}, rendering the method itself compatible to many different model architectures and (pre)processing steps. We will now perform a brief sanity check at hand of the linear model from Section 2.2 in order to demonstrate that the general decomposition rules derived for LRP carry the desired meaning.

Expressing the model from Equation (2.2) in terms of forward mappings, and expressing its bias as a generic mapping source (by assigning it to dimension 0 of the mapping input), we obtain an expression very similar to Equation (2.8)

$$f(\boldsymbol{x}) = z_f = \sum_d z_{df} \quad \text{with} \quad z_{df} = \begin{cases} w_d x_d & \text{if } d \geqslant 1 \\ b & \text{else} \end{cases} \tag{2.14}$$

which also covers the case of a multivariate output function f.

We can easily show that LRP, as given by Equation (2.13) and applied to the linear model, takes the form of its *inherent explainability* referred to earlier. We initialize $R_j^{(l+1)} = R_f = f(\boldsymbol{x})$ and and substitute z_{ij} with z_{df} to compute relevances R_d for input components x_d as

$$R_d = \sum_f \frac{z_{df}}{z_f} R_f = \sum_f \frac{z_{df}}{f(\boldsymbol{x})} f(\boldsymbol{x}) = z_{df} = \begin{cases} w_d x_d & \text{if } d \geqslant 1 \\ b & \text{else} \end{cases} . \tag{2.15}$$

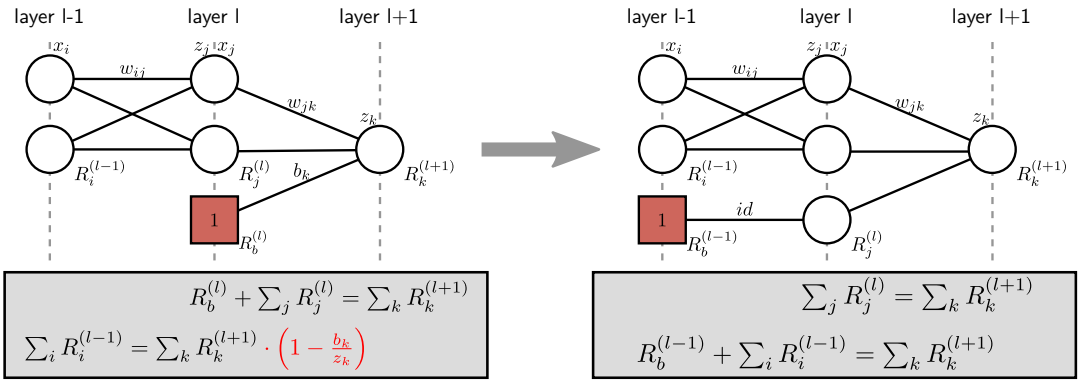

FIGURE 2.3: **Construction of a relevance conserving model from a model *leaking* relevance into bias nodes.** *Left:* A model with a bias node (red square) in layer l, absorbing some relevance $R_b^{(l)}$ and causing an inequality $\sum_i R_i^{(l-1)} \neq \sum_k R_k^{(l+1)}$. *Right:* If necessary, a functionally equivalent model strictly fulfilling Equation (2.5) can be constructed by treating the bias as a regular layer input and relocating its activation via an identity function id to previous layers.

Note that in the special case of a linear predictor, we have $z_{df} = R_{d \leftarrow f}$ and $z_f = f(\mathbf{x}) = R_f$[3] which is in general not true beyond the first step of decomposition. This can already be seen in the inequality $x_d \neq R_d$.

2.2.4 Relevance Conservation for Mapping Sequences with Bias Inputs

Many types of models such as Deep Neural Networks (see Section 2.3), Bag of Words prediction pipelines with Support Vector Machines (see Section 2.4) or the linear predictor decomposed in Section 2.2.3 use constant bias units as part of their prediction functions. The linear model in the previous section consists only of a single layer of computation and the bias b is treated as a constant input independent from given *sample* inputs $\mathbf{x}$. It has been shown that relevance is conserved between the output and the input of the model, *i.e.* between adjacent layers.

Let us now assume that the shown linear model is the final mapping of a prediction *pipeline* f and its input $\mathbf{x}$ depends on a series of preceding transformations of a given data point. As the bias b contributes to the output of the predictor, it absorbs a certain quantity of relevance R_b in proportion to its contribution to the predictor output (see Equation (2.3) in Section 2.1). Since b does not depend on the given input, the amount R_b is *stuck* in b and is not further propagated to previous layers of the pipeline. The relevances $R_{j \leftarrow k}$ directed from outputs k towards input-dependent x_j (and not b) – and thus preceding layers – then approximate the conservative properties of Equation (2.5). In particular,

$$\sum_j R_{j \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \cdot \left(1 - \frac{b_k}{z_k}\right) \text{ , s.t.: } j \text{ depends on model input}$$
$$R_{b \leftarrow k}^{(l,l+1)} = R_k^{(l+1)} \cdot \frac{b_k}{z_k} \,. \tag{2.16}$$

Since b actively contributes to f, an attribution of R_b to b is desired, despite preventing *exact* conservation of relevance from output to input. A *local* conservation of relevance between an upper layer neuron k and its inputs $\{j, b\}$, however, is given.

A functionally equivalent model can be constructed to strictly fulfill Equation (2.5) by moving the bias to the input layer and replacing it in its original and all preceding layers with an identity function proxy – *i.e.* by adding skip connections for the relocated bias – as shown in Figure 2.3. If necessary or meaningful[4], the residual bias relevance R_b can be redistributed

[3] more generally expressed: $z_{ij} = R_{i \leftarrow j}^{(l,l+1)}$ and $z_j = R_j^{(l+1)}$

[4] See the application of LRP to Bag of Words models in Section 2.4.2.

onto each input x_j of the decomposed layer. However, for models which are not purely sequential in structure such as the original AlexNet (Krizhevsky et al., 2012), ResNets (He et al., 2016; Zagoruyko et al., 2016) or DenseNets (G. Huang et al., 2017) a relaxation of the layer-wise conservation property in Equation (2.5) to a local relevance conservation property (Equation (2.9)) might be more consequential.

Adapting a perspective on LRP, as an algorithm operating on individual neurons (instead of whole layers) and their respective inputs, relaxes the idea of *layer*-wise relevance conservation to *local* relevance conservation and ensures $\sum_i R_i^{(1)} = f(\boldsymbol{x})$ for many non-sequential architectures via implicitly "reconstructing" models as shown in Figure 2.3.

2.2.5 Approximating Neuron Relevance via Taylor Decomposition

Consider a general function f whose pooling and activation does not fit into the structure given by Equation (2.10), and consequently intuition for a possible redistribution formula is lacking. Here we propose a strategy for such functions, based on a Taylor expansion of its outputs $z_j = f(\boldsymbol{x})$.

As f, consider a fully non-linear mapping function. Formula (2.11) demands backwards directed relevance messages to adhere to a pattern of weighting of output neuron relevances $R_{i\leftarrow j}^{(l,l+1)} = v_{ij} R_j^{(l+1)}$ for decomposition. For differentiable functions, such a weighting can be obtained by performing first order Taylor expansion wrt to a root point $\widetilde{\boldsymbol{x}}$, a free parameter which needs to be determined. A Taylor expansion of f around $\widetilde{\boldsymbol{x}}$ yields

$$f(\boldsymbol{x}) = \sum_{n=0}^{\infty} \frac{f^{(n)}(\widetilde{\boldsymbol{x}})}{n!} (\boldsymbol{x} - \widetilde{\boldsymbol{x}})^n \tag{2.17}$$

Assuming a multivariate input setting for the function f, we restrict the decomposition to a first degree Taylor series over partial derivatives wrt to each input component in order to construct a sum over (approximate) contributions to $f(\boldsymbol{x})$ for each input component i.

$$f(\boldsymbol{x}) \approx f(\widetilde{\boldsymbol{x}}) + \sum_i \frac{\partial f(\widetilde{\boldsymbol{x}})}{\partial x_i} (x_i - \widetilde{x}_i) \tag{2.18}$$

The elements of the sum can be assigned to incoming neurons and the zero-order term $f(\widetilde{\boldsymbol{x}})$ can be redistributed equally between them (should no $\widetilde{\boldsymbol{x}}$ satisfying $f(\widetilde{\boldsymbol{x}}) = 0$ be viable), leading to the decomposition

$$\forall_{i\leftarrow j} : z_{ij} = \frac{1}{n} f(\widetilde{\boldsymbol{x}}) + \frac{\partial f(\widetilde{\boldsymbol{x}})}{\partial x_i} (x_i - \widetilde{x}_i) \tag{2.19}$$

where n is the number of input components i for each j. Weightings v_{ij} satisfying Equation (2.11) can be obtained as

$$v_{ij} = \frac{z_{ij}}{z_j}\,, \tag{2.20}$$

leading to Equation (2.10). We call this process *Taylor Decomposition*, which constitutes a first step towards the framework of *Deep Taylor Decomposition* (DTD) (Montavon et al., 2016; Montavon et al., 2017). Within this thesis, Taylor Decomposition is used for the decompostition of Local Renormalization Layers (Krizhevsky et al., 2012) in Equation (2.45) in Section 2.3.1 and non-linear kernel functions which are not sum-decomposable in Equation (2.76) in Section 2.4.2.

For a more details on Deep Taylor Decomposition, the inclined reader can refer to (Montavon et al., 2016; Montavon et al., 2017).

2.2.6 Advanced Relevance Decomposition Rules

A drawback of the propagation rule of Equation (2.9) is that for small values z_j, relevances $R_{i\leftarrow j}$ can take unbounded values. In the following we introduce several adaptions based on

the decomposition rule of Equation (2.9) which are for one purposed to prevent numerical unboundedness and for another to achieve a desired effect. Table 2.1 provides an overview about the application of below decomposition rules in the following chapters.

TABLE 2.1: **Formulae and application of various LRP-rules throughout this thesis.**

LRP rule	Formula	Used for
ϵ-rule	$R_{i\leftarrow j} = \frac{z_{ij}}{z_j + \text{sign}(z_j)} \cdot R_j$	- DNN examples in Section 2.3.3 in parts together with the $\flat$-rule - BoW examples in Section 2.4.3 - DNN models in Chapter 3 - FV model in Chapters 4 and 6 - DNN models in Chapters 5 and 6, together with the $\alpha\beta$-rule and $\flat$-rule
$\alpha\beta$-rule	$R_{i\leftarrow j} = \left(\alpha\frac{z_{ij}^+}{z_j^+} + \beta\frac{z_{ij}^-}{z_j^-}\right) \cdot R_j$	- DNN models in Chapters 3 and 4 - DNN models in Chapters 5 and 6, together with the ϵ-rule and $\flat$-rule
$\flat$-rule	$R_{i\leftarrow j} = \frac{1}{\sum_{i'} 1} \cdot R_j$	- DNN examples in Section 2.3.3 in parts together with the ϵ-rule - BoW examples in Section 2.4.3 (implicitly) - FV model in Chapters 4 and 6 (implicitly) - DNN models in Chapters 5 and 6, together with the ϵ-rule and $\alpha\beta$-rule
w^2-rule	$R_{i\leftarrow j} = \frac{w_{ij}^2}{\sum_{i'} w_{i'j}^2} \cdot R_j$	- Weighted mapping connections, *e.g.* in DNNs (not applied throughout this thesis)

The ϵ-Rule

Unboundedness can be overcome by introducing a predefined stabilizer $\epsilon \geqslant 0$, yielding the ϵ-Rule:

$$R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} \frac{z_{ij}}{z_j+\epsilon} \cdot R_j^{(l+1)} & \text{if } z_j \geqslant 0 \\ \frac{z_{ij}}{z_j-\epsilon} \cdot R_j^{(l+1)} & \text{else} \end{cases}, \tag{2.21}$$

or, more concisely

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j + \epsilon \cdot \text{sign}(z_j)} \cdot R_j^{(l+1)}, \tag{2.22}$$

where the sign function requires adaption from $\text{sign}(0) = 0$ to $\text{sign}(0) = 1$ to avoid division by zero. However in that case, the conservation idea is relaxated further in order to gain better numerical properties. Considering the bias b explicitly as in Equation (2.16) the conservation law then becomes

$$\sum_i R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} R_j^{(l+1)} \cdot \left(1 - \frac{b_j+\epsilon}{z_j+\epsilon}\right) & \text{if } z_j \geqslant 0 \\ R_j^{(l+1)} \cdot \left(1 - \frac{b_j-\epsilon}{z_j-\epsilon}\right) & \text{else} \end{cases} \tag{2.23}$$

where we can observe that additional relevance is absorbed by the stabilizer. In particular, relevance is fully absorbed if the stabilizer ϵ becomes very large.

The $\alpha\beta$-Rule

An alternative stabilizing method that does not absorb further relevance consists of treating negative and positive pre-activations separately. Let

$$z_j^+ = \sum_i z_{ij}^+ \qquad\qquad z_j^- = \sum_i z_{ij}^- \tag{2.24}$$

where "$-$" and "$+$" denote the negative and positive part of the input transformations and bias contributions z_{ij}, *i.e.*

$$z_{ij}^+ = \begin{cases} z_{ij} & \text{if } z_{ij} \geqslant 0 \\ 0 & \text{else} \end{cases} \qquad\qquad z_{ij}^- = \begin{cases} 0 & \text{if } z_{ij} \geqslant 0 \\ z_{ij} & \text{else} \end{cases} . \tag{2.25}$$

This relevance propagation variant is defined as the $\alpha\beta$-Rule

$$\begin{aligned} R_{i\leftarrow j}^{(l,l+1)} = \left(\alpha\frac{z_{ij}^+}{z_j^+} + \beta\frac{z_{ij}^-}{z_j^-}\right) \cdot R_j^{(l+1)} \\ \text{s.t.} \quad \alpha + \beta = 1 \\ \alpha \geqslant 1 \end{aligned} \tag{2.26}$$

The $\alpha\beta$-rule allows to control manually the importance of positive and negative evidence, by choosing different factors α and β. Since $z_{ij} = z_{ij}^+ + z_{ij}^-$, enforcing $\alpha + \beta = 1$ ensures the relevance conservation property from Equation (2.5). This rule assumes that the output *activations* (in contrast to *inhibiting* outputs) of a layer are the result of positive logit activations. It thus suits the activation paradigm of Neural Networks with ReLU activation units especially well. A choice of $\alpha = 1, \beta = 0$ restricts the decomposition of the prediction to only activating forward mappings z_{ij}^+. In the framework of Deep Taylor Decomposition, this parameterization is referred to as the z^+-Rule for constrained input spaces (Montavon et al., 2016; Montavon et al., 2017). In later approaches, such as Excitation Backpropagation (J. Zhang et al., 2018), neural attention backpropagations are computed equivalently to the z^+-rule for CNN models with ReLU activation units. The philosophy of treating positive and negative inputs and forward activations separately for computing contributions of feature importance can also be found in the DeepLift (Shrikumar et al., 2016; Shrikumar et al., 2017) method.

The parameter settings $\alpha = 1, \beta = 0$ and $\alpha = 2, \beta = -1$ (which also incorporates inhibitory signals to the layer activation) are used for decomposing DNN predictions throughout this thesis.

The $\flat$-Rule

We use the $\flat$-rule (as in a "flat" distribution) as a heuristic means to project relevance values attributed to intermediate representations of a predicted sample onto the input. In practice, we set $\forall_{i,j} : z_{ij} = 1$ uniformly. The resulting decomposition rule then simply is

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{1}{\sum_{i'} 1} \cdot R_j^{(l+1)} , \tag{2.27}$$

which projects $R_j^{(l+1)}$, the relevance attributed to the upstream unit j, across all its inputs i (*i.e.* the *receptive field* of neuron j) uniformly. Here, index i' iterates over the same set of input units as i. Applying this rule to the input layer of a prediction model yields relevance attributions invariant to the normalization of the input, as demonstrated in Section 2.3.3. Alternative decompositions sharing this property are the w^2-rule discussed below or the z^B-rule (bounded rule) introduced via the Deep Taylor Decomposition framework (Montavon et al., 2016; Montavon et al., 2017). The $\flat$-rule presents a principled alternative to simply (up)scaling hidden neuron relevance for explaining the contributions of intermediate input representations, fitting the LRP framework. A more detailed introduction will follow in Chapter 5.

The w^2-Rule

An alternative to the $\flat$-rule which is invariant to shifts and normalizations in the input space, but makes use of the learned (Neural Network) model parameters is the w^2-rule:

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{w_{ij}^2}{\sum_{i'} w_{i'j}^2} \cdot R_j^{(l+1)} \tag{2.28}$$

Here, the index i' iterates over the same set of inputs as i. In (Montavon et al., 2016; Montavon et al., 2017), the w^2-rule has been derived for unconstrained input spaces, as opposed to the z^+-rule operating on inputs limited to take values from $\mathbb{R}^+$. The w^2-rule is part of the LRP framework but is not used in the evaluations presented within the scope of this thesis.

2.3 LRP for Deep Neural Networks

As we have already discussed in the introduction, Deep Neural Network (DNN) models can be considered the current state of the art for solving a wide variety of prediction problems.

However, until recently, DNNs and other complex, non-linear learning machines have been used and accepted as black-box models, providing little information about which aspect of an input causes the actual prediction beyond the generalization error. In most cases, DNNs do not rely on feature representations hand-crafted based on a-priori knowledge, but generally perform better when learning the optimal feature transformations for the problem settings on their own, as part of the training process, adding another layer of intransparency as a trade-off for increased prediction performance.

In the following, we will adapt the basic LRP decomposition rules from Equations (2.10) and (2.13) to layers commonly used in DNN architectures. The derived decompositions can be regarded as a basis, ready for adaption to the advanced rules proposed in Section 2.2.6, if necessary.

2.3.1 Decomposition of Neural Network Layers

The set of rules defining LRP can be integrated trivially with DNN architectures. Multilayer networks are commonly built as a set of interconnected neurons organized in a layer-wise manner. They define a mathematical function when combined to each other, that maps the first layer neurons (input) to the last layer neurons (output). We denote each neuron by x_i where i is an index for the neuron. By convention, we associate different indices for each layer of the network. We denote by "$\sum_i$" the summation over all neurons of a given layer, and by "$\sum_j$" the summation over all neurons of another layer (See Figure 2.2). A common mapping from one layer to the next one consists of a linear projection

$$\begin{aligned} z_{ij} &= x_i w_{ij} \\ x_j &= \sum_i z_{ij} + b_j \end{aligned} \tag{2.29}$$

or a (often component-wise) monotonously increasing non-linearity

$$x_j = \sigma(x_i). \tag{2.30}$$

The former (linear) case represents the commonly used fully connected (or dense) layers and convolutional layer types, where w_{ij} are the entries of the learned weight kernel connecting the neuron x_i to neuron x_j and b_j is a bias term, or certain types of pooling functions. In latter case, σ is a potentially non-linear activation function or normalization operator. In all cases, the indices i describe the set of input neurons x_i connected to x_j as inputs. Deep multilayer networks stack several of these functions, each of them, composed of a large number of neurons. Common non-linear functions are the hyperbolic tangent $\sigma(\mathbf{x}) = \tanh(\mathbf{x})$ or the rectifying linear unit (ReLU) $\sigma(\mathbf{x}) = \max(0, \mathbf{x})$. This formulation of Neural Network is general enough to encompass a wide range of architectures such as the simple Multilayer Perceptron (Rumelhart et al., 1986) or Convolutional Neural Networks (LeCun, 1998). That

is, relevance decomposition rules for most commonly used Neural Network layer types can be obtained by implementing Formula (2.10) from Section 2.2.

Linear Layers

Neural Network layers implementing linear transformations via weight kernels – that is, fully connected layers (dense layers) and all variants of convolution layers – implement a function as in Equation (2.29). Denoting with i input neurons and correspondingly connected output neurons with j, and assuming a given $R_j^{(l+1)}$, relevance decomposition can be implemented by taking z_{ij} from Equation (2.29) and choosing $z_j = x_j$, *i.e.* the layer's output activation. Decompsition into backwards-directed relevance messages then yields

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{x_i w_{ij}}{\sum_i x_i w_{ij} + b_j} \cdot R_j^{(l+1)} = \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)} \tag{2.31}$$

where the bias b is treated as a constantly activating and neuron connected to the outputs via weights b_j. The treatment of the bias as a regular neuron may in some cases cause noticable differences in the relevance sums of adjacent network layers due to the bias' absorption of relevance quantities in proportion to is contribution to the output activation. Section 2.2.4 discusses relevance conservation and leakage with bias-like model elements in detail.

Pooling Layers

Pooling layers such as sum-, average- and max pooling layers also readily fit the pattern of Formula (2.10).

Sum- and average pooling. Both average- and sum pooling operations can be generalized to convolution operations, with $w_{ij} = 1$ for the sum pooling layer and $w_{ij} = 1/n$ for the average pooling layer, where n is the number of neuron inputs.

Max pooling. Max pooling layers implement a function

$$x_j = \max_i(x_i) \tag{2.32}$$

for each output neuron j over all its inputs i. For a fixed neuron j, the associated relevance amount $R_j^{(l+1)}$ given as input for the relevance backward pass should thus be attributed to the (single) maximally activating input neuron x_i alone:

$$R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} R_j^{(l+1)} & \text{if } \arg\max_i(x_i) \\ 0 & \text{else} \end{cases} \tag{2.33}$$

To ensure local relevance conservation for the in practice unlikely case of multiple input neurons x_i maximally activating simultaneously wrt the same output neuron j, *i.e.* $|\{i|\max_i(x_i)\}| > 1$, the forward pass of the max pooling layer needs to be formulated alternatively to fully comply to the requirements imposed by LRP:

$$\begin{aligned} z_{ij} &= \begin{cases} \frac{x_i}{|\{i|\max_i(x_i)\}|} & \text{if } i \in \{i|\max_i(x_i)\} \\ 0 & \text{else} \end{cases} \\ x_j &= \sum_i z_{ij} \end{aligned} \tag{2.34}$$

The relevance backward messages then become

$$R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} \frac{R_j^{(l+1)}}{|\{i|\max_i(x_i)\}|} & \text{if } i \in \{i|\max_i(x_i)\} \\ 0 & \text{else} \end{cases} \tag{2.35}$$

Activation Layers

Within the transformations applied by commonly used component-wise activation layers such as tanh and ReLU, there is no mixing of input activations x_i to compute output activations x_j. In other words, representing with γ any generic component-wise function implementing an activation layer, the layer's forward pass computes

$$\begin{aligned} z_{ij} &= \delta_{ij}\gamma(x_i) \\ x_j &= \sum_i z_{ij} \,, \end{aligned} \tag{2.36}$$

where δ_{ij} is the Kronecker delta and thus the entries z_{ij} describe a diagonal matrix. The dimensionality of the input space of the activation layer is identical to the dimensionality of the output and the relationship between input dimension i and output dimension i is bijective. The relevance backward messages thus compute as

$$R_{i\leftarrow j}^{(l,l+1)} = \begin{cases} R_j^{(l+1)} & \text{if } \delta_{ij} \\ 0 & \text{else} \end{cases} \tag{2.37}$$

or, in short, $R^{(l)} = R^{(l+1)}$.

Merge Layers

Recent evolutions in DNN architectures have brought forth models with a higher degree of interconnectivity, *e.g.* with multiple feature transformations running in parallel as with the GoogleNet's inception modules (Szegedy et al., 2015), the ResNet architectures' skip connections (He et al., 2016; Zagoruyko et al., 2016) which are effectively shortcuts through the model which allow information to bypass certain (non-linear) transformations or the approach of DenseNet (G. Huang et al., 2017) to provide a layer's output as input to *all* succeeding layers.

Approaches for merging information used in above examples can be categorized into merge by aggregation (*e.g.* summation (He et al., 2016; Zagoruyko et al., 2016)) and merge by tensor concatenation wrt a certain tensor axis (Szegedy et al., 2015; G. Huang et al., 2017), for which relevance decomposition can be achieved easily.

Merge by aggregation. Layers merging outputs from multiple layers (or paths thoughout the network) by addition (or subtraction) implement a function

$$x_j = x_{1,j} + x_{2,j} + \cdots + x_{n,j} \tag{2.38}$$

where $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are the n input tensors assumed to be identical in dimensionality, which are to be merged component-wise at neuron indices j. As with component-wise activation functions, there is no mixing of information between the input neurons j. Hence, relevance decomposition functions analogously to the linear layer, with

$$\begin{aligned} z_{ij} &= x_{i,j} \text{ and} \\ x_j &= \sum_i z_{ij} \,. \end{aligned} \tag{2.39}$$

Equation (2.10) applies to compute the relevance backward messages without further adaption and $x_j = z_j$.

Merge by concatenation. Layers of neurons merged by concatenation along a certain tensor axis do not interact in any way, besides being stacked. The forward pass

$$x_j = [x_{1,j},\, x_{2,j},\, \ldots,\, x_{n,j}] \,, \tag{2.40}$$

with j indexing the neuron tensor's axes othorgonal to the concatenation axis is countered by an equivalently simple relevance decomposition rule

$$\left[R_{1,j}^{(l)},\ R_{2,j}^{(l)},\ \ldots,\ R_{n,j}^{(l)}\right] = R_j^{(l+1)} \tag{2.41}$$

Similar to the case of relevance conservation with bias-like model elements discussed in Section 2.2.4, the distinction between merge type layers and other layer types becomes obsolete when considering LRP on the basis of single neurons j and their relations to all its immediately preceding inputs i, instead of on the basis of layers of computation.

Normalization Layers

Normalization layers have been shown to improve the performance of DNNs (Krizhevsky et al., 2012) and to dramatically accelerate model training (Ioffe et al., 2015).

Local renormalization layers. Consider the local renormalization (Krizhevsky et al., 2012) z_j of a neuron x_j by the set of its surrounding neurons $\{x_k\}_k$ as

$$z_j = \frac{x_j}{\left(1 + b\sum_k x_k^2\right)^c} \tag{2.42}$$

where b and c are hyper-parameters determined during training on a validation set (Krizhevsky et al., 2012). This interaction can be modeled by a locally acting activation layer in the network, but can not be tackled exactly by LRP as introduced in (S. Bach et al., 2015) and Equations (2.8) to (2.12). However, the Taylor Decomposition strategy (Section 2.2.5, Equation (2.19)) can be applied.

One choice to be made is the point $\widetilde{x}$ at which to perform the Taylor expansion. There are two apparent candidates, firstly the actual input to the renormalization layer $\widetilde{x}_1 = x$ and, secondly, the input corresponding to the case when only the neuron j, which is to be normalized, fires: $\widetilde{x}_2 = (0,\ \ldots,\ 0,\ x_j,\ 0,\ \ldots,\ 0)$.

The partial derivative of z at $\widetilde{x}_2$ is zero for all variables x_i with $i \neq j$ due to

$$\frac{\partial z_j}{\partial x_i} = \frac{\delta_{ij}}{\left(1 + b\sum_k x_k^2\right)^c} - 2bc\frac{x_i x_j}{\left(1 + b\sum_k x_k^2\right)^{c+1}}, \tag{2.43}$$

which implies that the Taylor approximation of $z_j(x)$ (Equation (2.18)) around $\widetilde{x}_2$ has no off-diagonal contribution:

$$\begin{aligned} z_j(x) &\approx z_j(\widetilde{x}) + \sum_i \left(\frac{\delta_{ij}}{\left(1 + b\sum_k x_k^2\right)^c} - 2bc\frac{\widetilde{x}_i x_j}{\left(1 + b\sum_k x_k^2\right)^{c+1}}\right)\cdot(x_i - \widetilde{x}_i) \\ &\overset{\widetilde{x}=\widetilde{x}_2}{\approx} z_j(\widetilde{x}_2) + \sum_i \begin{cases} (\cdots)\cdot 0 & \text{if } i = j \\ 0 \cdot x_i & \text{else} \end{cases} \\ &\approx z_j(\widetilde{x}_2) + 0 = \frac{x_j}{\left(1 + bx_j^2\right)^c}. \end{aligned} \tag{2.44}$$

A Taylor expansion around $\widetilde{x}_2$ is thus equivalent to relevance attribution by identity, *i.e.* is equivalent to a weighting $v_{ij} = \delta_{ij}$.

We therefore apply the Taylor series around $\widetilde{x}_1$, yet since $x - \widetilde{x}_1 = 0$ we make use of Equation (2.44) and express $z_j(x)$ in terms of $z_j(\widetilde{x}_2)$ as an approximation around $\widetilde{x}_1$:

$$\begin{aligned} & z_j(\widetilde{x}_2) \approx z_j(\widetilde{x}_1) + \nabla z_j(\widetilde{x}_1)\cdot(\widetilde{x}_2 - \widetilde{x}_1) \\ \Longrightarrow\ & z_j(x) = z_j(\widetilde{x}_1) \approx z_j(\widetilde{x}_2) + \nabla z_j(\widetilde{x}_1)\cdot(\widetilde{x}_1 - \widetilde{x}_2) \\ \Longrightarrow\ & z_j(x) = z_j(\widetilde{x}_1) \approx \frac{x_j}{\left(1 + bx_j^2\right)^c} - 2bc\sum_{i:i\neq j}\frac{x_i^2 x_j}{\left(1 + b\sum_k x_k^2\right)^{c+1}} \end{aligned} \tag{2.45}$$

The resulting weighting

$$\nu_{ij} = \frac{1}{n}\frac{x_j}{\left(1+bx_j^2\right)^c} - \begin{cases} 2bc\frac{x_i^2 x_j}{\left(1+b\sum_k x_k^2\right)^{c+1}} & \text{if } i \neq j \\ 0 & \text{else} \end{cases} \tag{2.46}$$

satisfies the following qualitative properties: For neuron x_j which is to be normalized, the sign of the relevance is kept. For suppressing neighbouring neurons x_k with $j \neq k$, the sign of the relevance can be inverted in line with their suppressing property. The absolute value of the relevance received by the suppressing neurons is proportional to the square of their input. In the limits $c \to 0$ and $b \to 0$, the local renormalization converges against the identity and the approximations recovers the identity. A base line to compare against is to treat the normalization as a constant. In that case, the weights ν_{ij} for the relevance propagation in Formula (2.11) become a one-hot vector and relevance is propagated only to the neuron which is to be normalized: $\nu_{ij} = \delta_{ij}$. The work of (Binder et al., 2016b) verifies the weighting in Equation (2.46) as the superiour choice over the identity function quantitatively.

Batch normalization layers. Batch normalization layers (Ioffe et al., 2015) impose an affine transformation onto each mini-batch (as determined by that batch) passed through the layer during traing. Batch normalization counter-acts a phenomenon known as (internal) covariate shift (Shimodaira, 2000), which occurs when the distribution of the (DNN) model's (hidden) input activations changes, which in turn hampers model convergence. The use of Batch normalization is known to considerably speed up the training of DNN models, making the model less susceptible to weight initialization and can act as a regularizer (Ioffe et al., 2015). The layer implements a component-wise function

$$z = \frac{x - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}\gamma + \beta\,. \tag{2.47}$$

The γ and β are scale and shift parameters which are learned during model training and μ_B and $\sigma_{\mathcal{B}}^2$ are the mini batch mean and variances (as computed over the batch axis), which are adaptive to the input batches during training. After training, $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ are set to fixed values computed over the whole training population. The parameter ϵ is a small numerical constant to avoid divisions by zero. For brevity, we will substitute the scaling applied in Equation (2.47) as $s = \gamma \cdot (\sigma_{\mathcal{B}}^2 + \epsilon)^{-\frac{1}{2}}$

The application of a Batch normalization during inference can be seen as a sequence of (component-wise) additive and multiplicative terms.

$$x' = x - \mu_{\mathcal{B}}\,, \tag{2.48}$$
$$x'' = x' \cdot s\,, \tag{2.49}$$
$$z = x'' + \beta\,. \tag{2.50}$$

Both translations (Equations (2.48), (2.50)) can be considered as additive merge operations with constant bias tensors. Consequently, a relevance backward pass can be derived from the decomposition rules for additive merge-type layers (Equation (2.39)) and component-wise activation functions (Equation (2.37)) described earlier, as an alternative to an identity backward pass of $R^{(l+1)}$. The scaling operation in Equation (2.49) will, due to the bijective input-output neuron interaction (akin to activation layers), cancel out in the relevance backward pass.

$$\text{decomposing (2.50)} \;:\; R_{x''} = x''\frac{R^{(l+1)}}{z} \;;\; R_\beta = \beta\frac{R^{(l+1)}}{z} \tag{2.51}$$
$$\text{decomposing (2.49)} \;:\; R_{x'} = R_{x''} \tag{2.52}$$
$$\text{decomposing (2.48)} \;:\; R^{(l)} = x\frac{R_{x'}}{x'} \;;\; R_{\mu_{\mathcal{B}}} = \mu_{\mathcal{B}}\frac{R_{x'}}{x'} \tag{2.53}$$

The variables $R_{x'}, R_{x''}, R_{\mu_{\mathcal{B}}}$ and R_β reflect the path of the backward directed relevance and where certain amounts of relevance can be absorbed by bias-like variables $R_{\mu_{\mathcal{B}}}$ and R_β (see

Section 2.2.4). A compiled expression for the relevance backward pass through Equations (2.50) to (2.48) yields

$$R^{(l)} = \frac{x \odot x'' \odot R^{(l+1)}}{x' \odot z} = \frac{x \odot s \odot R^{(l+1)}}{z} \tag{2.54}$$

where the operator $\odot$ stands for the Hadamard product for element-wise multiplication. Note that the parameters β and γ are regarded as optional due to their redundancy in certain combination of Neural Network layers (Rasmus et al., 2015). For such models, the adapted forward pass must be considered in the relevance backward pass.

The Softmax Output Layer

The softmax function is commonly used as the last layer for Neural Network architectures, especially when training under a cross-entropy loss regime. The function takes a vector of real values z (indexed by i and i') and transforms them into a vector of probabilities summing to 1, while maintaining in its outputs the ranking order of its inputs:

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{i'} e^{z_{i'}}} \tag{2.55}$$

Similar to (Simonyan et al., 2013), we discourage from using a Softmax output layer with LRP, *e.g.* by simply removing it from the model when applying LRP, for the following reasons:

(1) The softmax function computes the probabilistic mapping wrt to *all* given logit activations z, hindering the individual assessment of model outputs. Assume a model computing predictions in $\mathbb{R}^2$, indicating the presence of a class if the outputs are (strongly) positive. The model is given two inputs x_1 and x_2 provoking the (logit) predictions $z_1 = [1, 1]$ and $z_2 = [1, 0]$ respectively. The application of the softmax function correspondingly yields $\text{softmax}(z_1) = [0.5, 0.5]$ and $\text{softmax}(z_2) = [0.73\ldots, 0.26\ldots]$. Now, despite the logit outputs $z_{1,1}$ and $z_{2,1}$ activating identically, their softmax transformations deviate due to the dependency to other dimensions of the logit vectors. The softmax function thus obfuscates the *actual* confidence of the model's prediction of each class, making comparative analysis with LRP between samples and output neurons difficult when the softmax layer is not removed from the model.

(2) The softmax function is invariant to constant shift. Given an arbitrary scalar c added to each dimension of the logit vector z,

$$\text{softmax}(z + c)_i = \frac{e^{z_i + c}}{\sum_j e^{z_j + c}} = \frac{e^c e^{z_i}}{e^c \sum_j e^{z_j}} = \frac{e^{z_i}}{\sum_j e^{z_j}} = \text{softmax}(z)_i \,. \tag{2.56}$$

Together with (1), this can become deceptive when passing off-manifold (or "trash") inputs to the model yielding low confidence logit outputs for all classes, *e.g.* $z_3 = [-1, -2] = z_2 + [-2, -2]$. The application of $\text{softmax}(z_3) = [0.73\ldots, 0.26\ldots]$, which is identical to $\text{softmax}(z_2)$ despite the difference in z_2 and z_3. This invariance property further hinders the comparative analysis between samples and predictions for target classes.

(3) The softmax function acts as a terminal non-linear activation function, mapping all negative inputs from a given z into $\mathbb{R}^+$, with no following linear mapping and decomposition steps. The function can be regarded as a element-wise application of exp, followed by an ℓ_1-normalization step. Since both mapping steps operate on individual input components, the corresponding LRP backward pass would be the identity function. Assuming a negative logit value for a model output of interest, an initialization with the output of the following softmax layer (which is always positive) would, due to the identity backward pass through the softmax layer, impose $R^{(\text{init logit})} = -c \cdot R^{(\text{init softmax})}$, where $c \in \mathbb{R}^+$ is some scaling applied to transform the logit output to the softmax output of the model. In other words, the sign of every unit of relevance attributed throughout the network would disagree between relevance scores computed wrt to the logit output compared to relevance scores compared to the softmax output. The same problem applies to other activation functions used at the

model output not satisfying $\gamma(0) = 0$, such as the logistic sigmoid function $\frac{1}{1+e^{-x}}$. A possible solution to (3) could be found in an approximate Taylor decomposition approach. Here, however, a root point $\widetilde{x}$ satisfying $f(\widetilde{x}) = 0$ does not exist for the softmax and the logistic sigmoid.

We therefore suggest to remove any (non-linear) activation functions terminating the prediction function not satisfying $\gamma(0) = 0$ from the model before the application of LRP for reason (3), and the softmax function for reasons (1) and (2) especially.

2.3.2 Efficiently Implementing LRP for Neural Networks as a Modified Gradient Backward Pass

Recent work has claimed identity between "gradient times input" and LRP (Shrikumar et al., 2016), which is not true in general. The works of (Kindermans et al., 2016; Ancona et al., 2018) however provide proof that the basic decomposition rule from Equation (2.13) – assuming only ReLU activations and max-pooling nonlinearities, the consideration of the bias as regular neuron and absent further modifications (*e.g.* purposed decomposition rules as in Section 2.2.6) – is equivalent to "gradient times input" or an element-wise product between the model input and the derivation of the prediction function. In this section, we provide further details on the similarities and differences between the "gradient times input" approach and LRP and show how relevance decompositions can be implemented for Deep Neural Networks as modified gradient backward passes.

Assuming a network with layered architecture and activation functions σ following each linear mapping (*e.g.* dense layers, convolutional layers), the forward pass from layer (l) to layer $(l+1)$ can be expressed as

$$x^{(l+1)} = \sigma\left(W^T x^{(l)} + b\right) = \sigma\left(z^{(l+1)}\right) \tag{2.57}$$

where $x^{(l)}$ are the inputs at layer (l), $z^{(l+1)}$ are the preactivations after the application of the linear transformation $\{W, b\}$ and $x^{(l+1)}$ are the non-linearly activated outputs. Assuming top layer relevances $R^{(l+1)}$ corresponding to the neurons of $x^{(l+1)}$, the relevance for $x^{(l)}$ can be computed as

$$R^{(l)} = \underbrace{x^{(l)}}_{\text{input}} \odot \underbrace{\frac{\partial z^{(l+1)}}{\partial x^{(l)}}}_{\text{gradient}} \cdot \underbrace{\frac{R^{(l+1)}}{z^{(l+1)}}}_{\text{scaled relevance}} \tag{2.58}$$

according to the relevance decomposition rules for linear (Equation (2.31)) and activation layers (Equation (2.37)) in Section 2.2.6, as the relevance backward pass for component-wise activation function is resolved via the identity function. What remains is "gradient times input" for the linear mapping component multiplied with an element-wise scaling of the top layer relevances.

Connecting LRP decomposition steps for consecutive layers following above decomposition yields

$$\left.\begin{aligned} R^{(l)} &= x^{(l)} \odot \frac{\partial z^{(l+1)}}{\partial x^{(l)}} \cdot \frac{R^{(l+1)}}{z^{(l+1)}} \\ R^{(l+1)} &= x^{(l+1)} \odot \frac{\partial z^{(l+2)}}{\partial x^{(l+1)}} \cdot \frac{R^{(l+2)}}{z^{(l+2)}} \end{aligned}\right\} R^{(l)} = x^{(l)} \odot \frac{\partial z^{(l+1)}}{\partial x^{(l)}} \cdot \underbrace{\frac{x^{(l+1)}}{z^{(l+1)}}}_{\text{nonlinearity}} \odot \frac{\partial z^{(l+2)}}{\partial x^{(l+1)}} \cdot \frac{R^{(l+2)}}{z^{(l+2)}} \tag{2.59}$$

which resolves the non-linearities between layers as

$$\frac{x^{(l+1)}}{z^{(l+1)}} = \frac{\sigma(z^{(l+1)})}{z^{(l+1)}}, \tag{2.60}$$

the element-wise output-input ratio of the non-linear mapping σ. For linearly activating

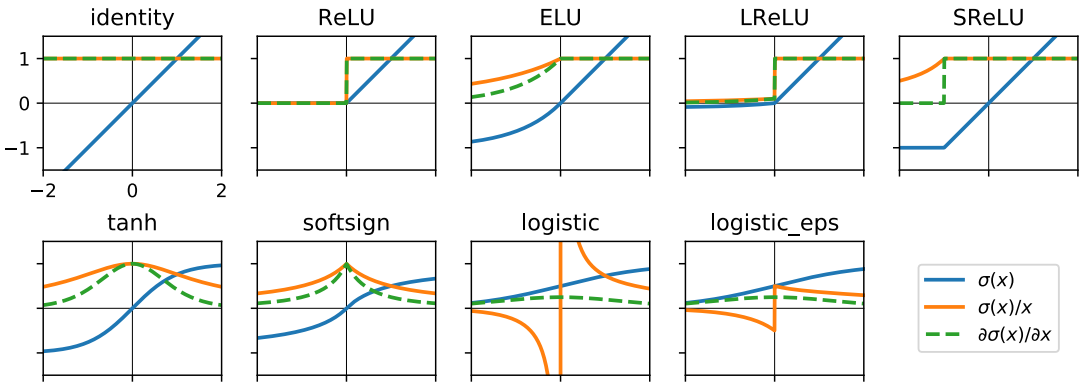

FIGURE 2.4: **Qualitative comparison of the behaviour of activation functions σ to its derivative and its output-input ratio as used in LRP.** The blue line shows the each function value $\sigma(x)$ over a range of $x \in [-2, 2]$. The dashed green line is the function's derivative $\partial\sigma(x)/\partial x$ at x and the solid orange line its output-input ratio $\sigma(x)/x$. All figures are scaled identically. *Top row:* (Piece-wise) linear activation functions such as variations of the ReLU (Clevert et al., 2016). *Bottom row:* Fully non-linear activation functions and the figure legend. The output-input ratio allows LRP to backpropagate relevances $R^{(l+1)}$ to previous layers, where the backward gradient already blocks backward directed signals, *e.g.* for the tanh function due to the saturation of $\sigma(x)$ of the firing neuron causing $\partial\sigma(x)/\partial x \approx 0$. For the family of logistic functions not satisfying $\sigma(0) = 0$ the application of numerically stabilizing rules – *e.g.* the ϵ-rule as applied in the curve "logistic_eps" – to the preceding linear mapping layer is recommended: A $z_j \approx 0$, *e.g.* caused by positive and negative mappings z_{ij} cancelling each other out, is mapped to $\sigma(z_j) \approx 0.5$ by the logistic sigmoid function. The non-linearly activated neuron thus may receive non-zero $R_j^{(l+1)}$. A scaling via z_j without further stabilization may cause unbounded numerical values. LRP incorporates the sign switch of the logistic function at $x = 0$ for $x < 0$ into the scaling of relevance, achieving meaningful attributions of relevance to lower layers following the activation tendency of z_j via the forward mappings z_{ij}.

non-linearities such as the ReLU[5] this ratio becomes 1. In that special case, an LRP decomposition of a k-layer model is equal to "gradient times input", since LRP initializes its (selected) output neurons with $R^{(k)} = z^{(k)} = f(x)$:

$$R = \underbrace{x \odot \frac{\partial z^{(1)}}{\partial x} \cdot \frac{\sigma(z^{(1)})}{z^{(1)}} \odot \ldots \odot \frac{\partial z^{(k)}}{\partial x^{(k-1)}} \cdot \frac{R^{(k)}}{z^{(k)}}}_{\text{LRP}} \overset{\frac{\sigma(z)}{z}=1}{=} \underbrace{x \odot \frac{\partial z^{(1)}}{\partial x} \cdot \ldots \cdot \frac{\partial z^{(k)}}{\partial z^{(k-1)}}}_{\text{"gradient times input"}} = x \odot \frac{\partial z^{(k)}}{\partial x} \tag{2.61}$$

Figure 2.4 qualitatively compares the behaviour of the backward gradient of a range of activation functions to the output-input ratio used in LRP.

Above derivations implicate that the decompositions performed with LRP are analytically meaningful (Kindermans et al., 2016) and that LRP and its rule adaptions (Section 2.2.6) can be implemented efficiently as modified gradient backward passes, as done in (Alber et al., 2018a; Alber et al., 2018b)[6]. Thus, LRP has a run time complexity of $O(f(x))$ for DNNs.

A key observation from Equations (2.59) and (2.60) is that the decomposition steps defining relevance propagation (Equations (2.10) and (2.12)) are invariant against the choice of function σ for computing relevances for the inputs $x^{(l)}$, conditioned on keeping the value of relevance $R^{(l+n)}$ for $x^{(l+n)}$ (for some $n \geqslant 1$) fixed. This observation allows to deal with any

[5] The derivative of the ReLU function is 1 for inputs $\geqslant 0$ and 0 otherwise, which "blocks" the backward gradient. See (Kindermans et al., 2016).

[6] The *iNNvestigate* software package for Neural Network analysis is available from https://github.com/albermax/innvestigate.

non-linear activation function σ. The function σ does however exert influence on computing the relevance $R^{(l)}$ by its influence on the relevance $R^{(l+n)}$ for $\mathbf{x}^{(l+n)}$. This can be explained by the fact, that the choice of σ determines the value of $\mathbf{x}^{(l+n)}$ and thus also relevance $R^{(l+n)}$ for $\mathbf{x}^{(l+n)}$ which gets assigned by the weights in the layers above. We remark again, that even max pooling fits into this structure as a limit of generalized means, see Equation (2.82) in the following Section 2.4 for example. For structures with a higher degree of non-linearity, such as local renormalization (Pinto et al., 2008; LeCun et al., 2010), Taylor approximation applied to neuron activations $\mathbf{x}^{(l+n)}$ can be used again to achieve an approximation for the structure as given in Equation (2.57).

Finally, it can be seen from the formulas established in this Section that LRP is different from a Taylor series or partial derivatives (times input). Unlike Taylor series, it does not require a second point other than the input image. Layer-wise (or neuron-wise) application of the Taylor series (Montavon et al., 2016; Montavon et al., 2017) can be interpreted as a generic way to achieve an approximate version of LRP. Similarly, in contrast to any methods relying on derivatives, differentiability or smoothness properties of neuron activations are not a necessary requirement for being able to define formulas which satisfy Layer-wise Relevance Propagation. In that sense it is a more general principle, applicable to a wider range of predictors. See *e.g.* Section 2.4, where LRP is applied to SVMs and Bag of Words pipelines.

2.3.3 Example Application on MNIST Data and Neural Networks

We provide an example application of LRP with three simple Neural Network architectures trained to predict hand written digit classes on the popular MNIST (LeCun, 1998) data set, to foster the intuition and understanding of relevance decomposition. The content of the MNIST data set is intuitively understandable by humans, yet complex enough to benefit from (highly) non-linear prediction functions for digit class differentiation (LeCun, 1998).

Example Network Architectures

In the following, we perform LRP on the LeNet-5 (LeCun, 1998) Convolutional Neural Network architecture and two fully connected Neural Networks used in (S. Bach et al., 2015), named MLP (for Multi-Layer Perceptron) for brevity. The MLP models consists of a sequence of three fully connected linear layers with 1296 hidden units each, which are either followed by tanh or ReLU activation functions (one type per model). A final linear output layer is mapping to the 10 outputs representing the MNIST digit classes. The LeNet-5 model consists of layer groups of ReLU-activated convolution layers, followed by an average pooling layer each. This structure repeats three times, until the output is reached. The trained models are available with the LRP Toolbox (Lapuschkin et al., 2016b).

The MLP models receive the (28×28) shaped MNIST samples as 784-dimensional vectors. Correspondingly, relevance values computed via LRP will also be available in vector form. For later visualization, the vectors of relevance scores can simply be reshaped into square images again. The LeNet-5 directly receives as input square image matrices, zero padded to a $(32 \times 32 \times 1)$ shaped input tensor with an additional channel axis.

All models have been trained using a standard Stochastic Gradient Descent (SGD) algorithm (LeCun et al., 2012), with input samples being translated up to 4 pixels in either direction as a means for data augmentation, and random mini batches of 25 samples with pixel values normalized to be within $[-1, 1]$. The MLP (tanh) and MLP (ReLU) models and the LeNet-5 achieve a generalization performance of 99.16%, 99.17% and 99.23% on the MNIST test set, respectively.

Colour Space Mapping for Human Interpretability

Relevance maps at the input layer of a classifier have the same dimensionality as the input sample $\mathbf{x}$. For prediction tasks in (colour) image recognition, this will usually result in a very large $(H \times W \times C)$ shaped tensor of relevance scores for all voxels, where H and W are the image height and width respectively and C is the number of channels, *e.g.* $C = 3$ for standard rgb images, which makes interpretation by a human observer difficult. For multi channel colour images, we can either observe each colour channel separetely or aggregate

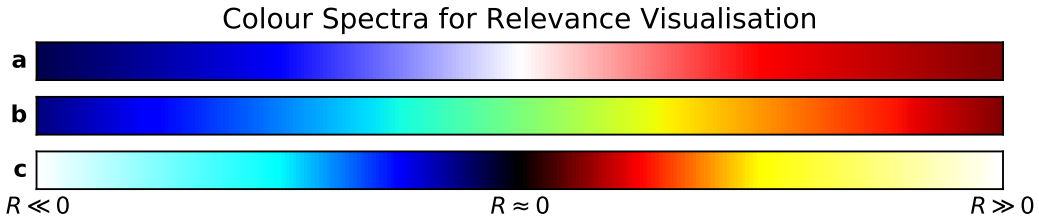

FIGURE 2.5: **Example colour space embeddings.** Colour mappings used throughout this thesis have symmetric and diverging properties for relevance map visualisation. In general, prediction-neutral relevance values ($R \approx 0$) are projected onto the center of the colour spaces, while negative relevances ($R < 0$) are represented as cold hues and positive relevances ($R > 0$) are shown in warm colours. Colour map **a** corresponds to the `seismic` colour map as built into python's `matplotlib` (Hunter, 2007) package. Colour map **b** is available in both `matplotlib` and `matlab` (MATLAB, 2016) as the `jet` colour map. Variant **c** is a custom implementation with increased contrast and is available as part of the LRP Toolbox (Lapuschkin et al., 2016b).

the relevance scores over the colour channel axis[7]. Throughout this thesis we chose the latter approach, to obtain a single relevance quantity per pixel coordinate, should subpixel relevance attributions exist.

To aid human experts in the assessment of relevance maps, we use colour space mapping to visualize the quality of the relevance attributed to each input pixel. Depending on the task of the model, relevance maps fulfill different properties. For a classification task, we can assume that $R_i \approx 0$ indicate input components neutral to the prediction of the data point, *i.e.* , the concept to detect is absent in the input variable i. Other than that, we know that $f(\boldsymbol{x}) = \sum_i R_i$ at input level. However, the individual R_i are unbounded. For classification, we therefore normalize relevance maps as

$$\forall i : R_i \leftarrow \frac{R_i}{S} \text{ with } S = \max_{i'}(|R_{i'}|) \tag{2.62}$$

which preserves $R_i = 0$ after the mapping and otherwise scales to $\forall i : R_i \in [-1, 1]$, which facilitates the controlled mapping to some colour space of choice. We suggest the use of diverging and symmetric colour spaces of high contrast which are both suitable for visualization on electronic displays and on print media. Choices for colour maps used throughout this thesis are shown in Figure 2.5.

Interpreting relevance maps

We begin with prediction tasks for all three models, on a number of randomly selected samples. Until stated otherwise, LRP is performed respectively the model decision, *i.e.* the dominantly firing output neuron. The computed pixel-wise relevance values are then mapped into a colour space. Figure 2.6 shows the input digits passed to the different neural networks, with the models' decisions' corresponding relevance maps below.

For the shown input digits, all models predict correctly. All the Neural Networks' functions encode predicted class membership as high positive output responses in the corresponding neurons. In the visualized relevance maps, this reflects in warm hues (standing for $R > 0$) identifying input components supporting the model prediction and cold hues (mapped from $R < 0$) pointing out evidence in the input considered as contradictory to the learned class (prototypes) by the model.

We can observe that – albeit all three models predicting with similar accuracies on the test set – that prediction strategies differ. Take as an example the digit "7" in the leftmost column of Figure 2.6: Both the MLP with ReLU units and the LeNet-5 predict correctly because they observe the top horizontal line of the digit and partly due to the diagonal "leg" of the digit shown in the input sample. Also note that white background pixels next to the drawn digit

[7] Meaningful aggregation of relevance values as a desired property will be discussed in Chapter 3, Section 3.3.

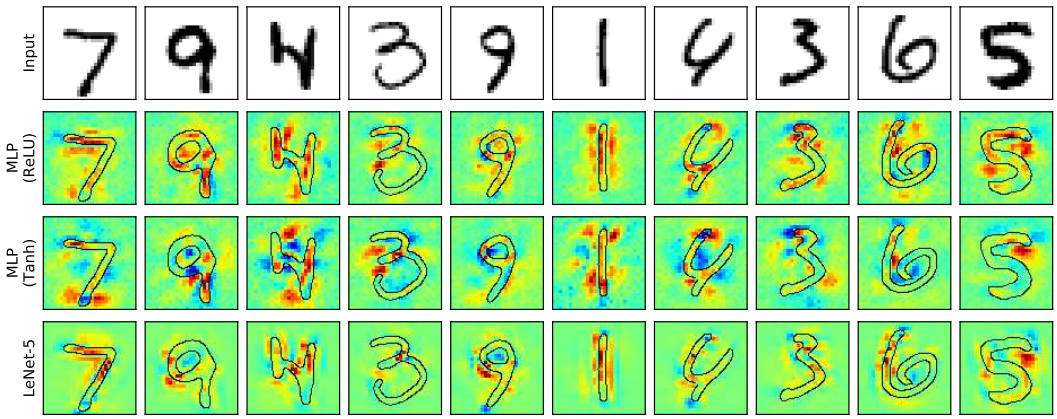

FIGURE 2.6: **Input samples and relevance maps for different Neural Networks.** *Top row:* Randomly selected MNIST inputs. 2*nd to* 4*th row:* Relevance maps wrt the models' dominantly firing output neurons per sample for the ReLU-activating MLP, the tanh-activating MLP and the LeNet-5 model. All samples are predicted correctly.

are considered positively relevant evidence; the digit is recognized due to the slimness of the line. Thicker black lines could represent crudely drawn digits "8" or "0". Compared to the LeNet-5 model, the MLP uses background information more extensively.

The MLP model with tanh activation units uses an entirerely different strategy for this input digit: It identifies the digit as a "7" due to the length of the top horizontal line – distinguishing that part of the image from a diagonally drawn "1" – but primarily due to the absence of a bottom horizontal line, which could make the input sample look more similar to a "2" if present. The last feature is in part shared in the prediction strategy used by the ReLU-activating MLP.

Negative relevances shown in blue colour correspond to input features weighting the model decision against the explained output. For all three models, the input digit "7" constutites a almost stereotypical representative for its class, according to the relevance response. Negative evidence shows, that for the ReLU-activated MLP model, the top bar of the digit is expected to be longer. The tanh-activated MLP seems to expect a seven as it is commonly written *e.g.* in central european countries, with a middle horizontal stroke, *i.e.* 7̸ . In Chapter 3, we verify qualitatively that relevance maps computed via LRP best represent the *decision making* of the model best, among heatmaps from other methods.

Output Neuron Selection and Relevance Map Normalization

The application of LRP is not restricted to a model's actual prediction. In a multi class setting, as it is the case with most DNN-based applications, LRP can be configured to decompose the prediction for any class output. In such a setting, the joint normalization of multiple relevance maps might desirable: Prior to normalization for colour space mapping, the relevance values R_i encode the prediction of the classifier $f(\mathbf{x})$ in both its composition and magnitude. Relevance maps can always be normalized individually, but it might be reasonable to normalize multiple relevance maps by a common factor S (see Equation (2.62)) derived from all relevances over all considered specimen, *e.g.* for comparing relevance responses for multiple samples wrt to the same target class or a single sample wrt to multiple target classes, by considering the differences in model output. Figure 2.7 presents relevance maps for each of the ReLU-activated MLP's output neurons for digit "7" from Figure 2.6, visualized individually or via a common scaling factor S.

Computing relevance maps for non-dominantly firing output neurons can reveal interesting information about the learned rejection strategy of the model, *i.e.* why a certain class has *not* been picked for prediction. The relevance map for output neuron 1 in Figure 2.7 shows, that the neural network model attributes positive relevance to the diagonal line of the digit, resembling a digit "1". However, the input digit has properties, which suppress the

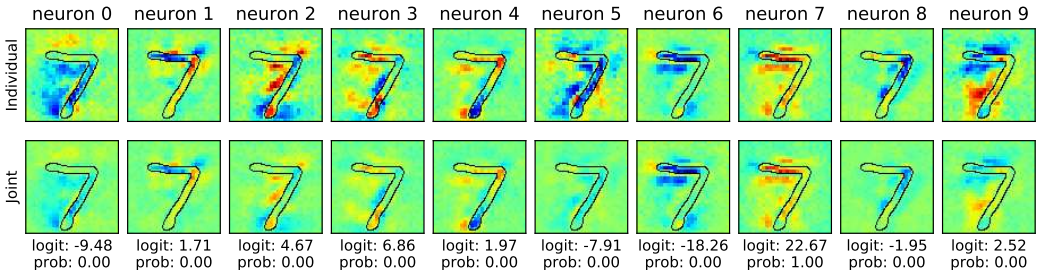

FIGURE 2.7: **Relevance maps for each output neuron of the MLP model with ReLU activations for input digit "7" from Figure 2.6.** *Columns:* Relevance maps wrt the (class representing) output neurons as identified by the column headers. *Top row:* Relevance maps normalized individually prior to colour mapping. *Bottom row:* Jointly normalized relevance maps with a common scaling factor S. At the bottom of each column, the value *logit* describes the pre-softmax neuron output and *prob* the corresponding post-softmax probabilistic mapping.

activation of the output neuron. This reflects in the negative relevance on the horizontal line at the digit's top, indicating *contradicting features* for the analyzed model output. Similarly intuitive relevance responses can be seen for other output classes, *e.g.* "2" and "3" which both share similar digit features with the correct output neuron "7".

While individually normalizing relevance maps for visualization (top row of Figure 2.7) increases contrast in colour space and can aid in the qualitative analysis of individual prediction outputs, a joint normalization (bottom row of Figure 2.7) preserves the relative magnitude of the decomposed output neuron activations. Referring to above example figure, we see that digit classes "0" and "5" do share almost no digit features with the provided class input, which reflects in strong negative relevance in the figure's top row of relevance maps. A joint normalization of all the neuron's relevance maps however reveals that the negative relevance responses for neurons "0" and "5" distribute more globally, indicating a global lack of the *right* input features for activating the respective neurons, while the relevance response for digit "1" for example is much more concentrated on fewer, more distinctly contradicting input components, despite the comparatively lower logit magnitude of neuron 1.

This property – a result of the relevance conservation property of LRP (Equation (2.5)) – is used in Section 2.4.3, Figure 2.13 for merging relevance from many image patch-wise predictions into one single larger relevance response map and is in general a desired property for aggregating and comparing relevance maps (see Chapter 6), especially for non-overlapping input patches.

In some cases selecting multiple output neurons for relevance back propagation instead of single neurons can be beneficial. Suppose a model trained on the hierarchical label set of the ImageNet (Russakovsky et al., 2015), for example. Here, one might be interesting in what makes the model interpret an input image as "frog" in general, instead of the sub categories "bullfrog", "tailed frog" or "tree frog" individually. In such cases, the simultaneous relevance backwards pass for multiple neurons will yield a weighted combination of relevance responses from all output classes of interest.

In general, however, explaining all output neurons representing $f(\mathbf{x})$ at once will yield nonsensical and unspecific relevance maps. Figure 2.8 demonstrates the relevance map for input digit digit "7" from Figure 2.6 as a simultaneous response from all output neurons at the same time.

The figure also demonstrates that the relevance response wrt all output neurons at once is equal to the sum over relevance maps wrt to all individual output neurons due to the relevance conservation property (Equation (2.5)). The observable deviations between the logit output in Figure 2.7 and input level relevance sums $\sum_r R_i$ in Figure 2.6 are caused by the attribution of relevance quantities R_b to the bias neurons of hidden layers, which are not further propagated towards the input. For details, see Section 2.2.4.

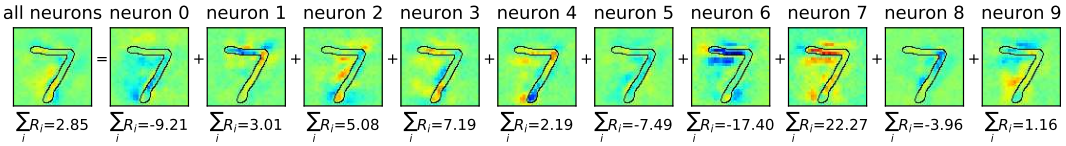

FIGURE 2.8: **Relevance response for input digit "7" from Figure 2.6 for all model output neurons at once.** This figure demonstates relevance conservation when applying LRP without selecting specific output neurons.

The Influence of Input Normalization to LRP and DNNs

As a method based on model-input interaction, LRP inherently depends on choices made for the input in normalization, shift and scale, via the analyzed model itself. Thus, we can use LRP to visualize the Neural Network's reaction to inputs under different normalization schemes. We follow the experimental protocol from (Kindermans et al., 2017a) to demonstrate the effect a change within the input range has on the network, reflecting in relevance explanations. Similar to (Kindermans et al., 2017a) we train a Neural Network classifier (we re-use the LeNet-5 architecture from earlier) to predict digit classes on MNIST, with pixel values normalized to a range of $\mathbf{x}_1 \in [0, 1]$. Here the value 0 encodes image background and 1 the foreground showing the digit. That first model operating on inputs from $\mathbf{x}_1$ is called "network A" and achieves a prediction accuracy of 98.71% on the test set. We follow (Kindermans et al., 2017a) in the construction of a second dataset $\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{m}$ with $\mathbf{m} = -\mathbb{1}$, *i.e.* $\mathbf{x}_2 \in [-1, 0]$ where -1 encodes background pixels and 0 encodes the foreground. A second, modified network ("network B") is then constructed from the first, which operates on the input domain $\mathbf{x}_2$.

The modified network is identical to the first one operating on $\mathbf{x}_1$, with the exception of the bias in the input layer, which has been adapted to compensate for the constant shift $\mathbf{m}$:

$$\begin{aligned} &\text{network A input layer: } \; z = W\mathbf{x}_1 + b \\ &\text{network B input layer: } \; z = W\mathbf{x}_2 + b_2 = W\underbrace{(\mathbf{x}_1 + \mathbf{m})}_{\mathbf{x}_2} + \underbrace{(b \overbrace{-W\mathbf{m}}^{\text{compensator}})}_{b_2} \end{aligned} \tag{2.63}$$

Thus, the activations z of the first (and all succeeding) neurons of both networks are identical. A graphical representation of the construction of both input layers can be found in Figure 2.9.

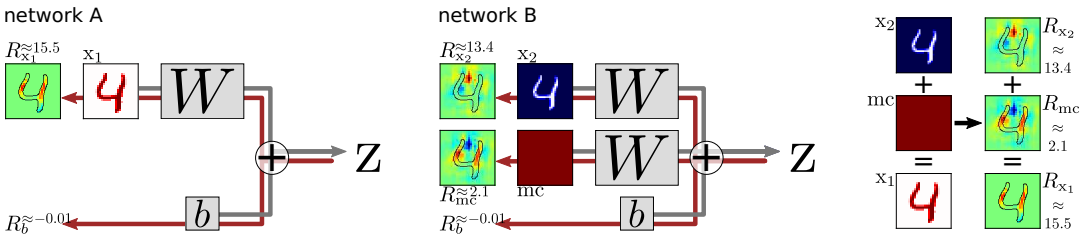

FIGURE 2.9: **Input layer of the trained "network A" and the constructed "network B" operating on inputs subject to a constant shift.** Gray arrows show the flow of forward activations z_{ij} and red arrows show the flow of backward directed relevance messages $R^{(1,2)}_{i \leftarrow j}$. To better visualize the samples from $\mathbf{x}_1$ and $\mathbf{x}_2$ in their respective ranges, we adapt the `seismic` color map and align its spectrum to a range of $[-1, 1]$ for the digit inputs. The resulting mapping visualizes values close to 0 as white color, positive values in red hues and negative pixel values in blue hues. The relevance attributed to the bias b does not change between both models. However, from network A to B a certain quantity of attributed relevance is shifted from $\mathbf{x}_1$ to $\mathbf{x}_2$ *and* $\mathbf{m}$ (and thus the bias b_2 from Equation (2.63)).

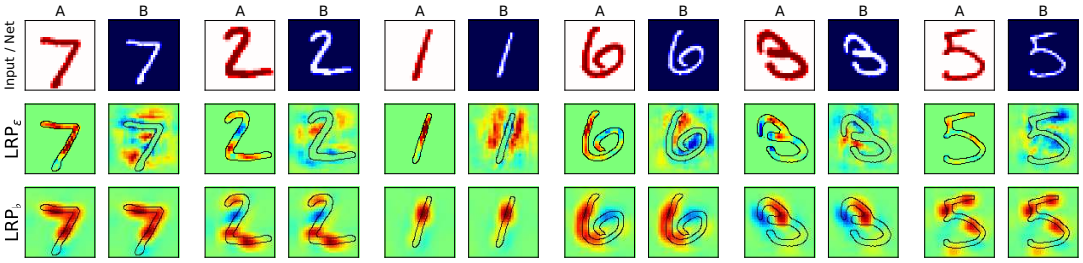

FIGURE 2.10: **With the appropriate choice of decomposition rules, LRP can be rendered invariant to transformations in input space.** Each colum of 3×2 images shows input samples (top row) and relevance maps for the sample input for both networks A and B when applying the ϵ-rule (center row) or the $\flat$-rule (bottom row). Decompositions performed with the $\flat$-rule do only depend on the passed down relevance values and each neuron's receptive field. Thus are invariant to both the inputs and the model parameters. The computed relevance attributions reflect neuron relevance of intermediate representations of the data, which are by construction identical between both networks. Note however, that for fully connected (dense) layers, the application of the $\flat$-rule does yield uniform relevance maps, since each output neuron of such layer receives inputs from all preceding neurons.

When applying LRP to both models for (except for the applied shift) identical samples we can immediately spot striking differences in the attribution of relevance to the given input pixels, as shown in Figure 2.9. For samples from $\mathbf{x}_1$ and network A, relevance is only attributed *on the digit itself*. Since background pixels in $\mathbf{x}_1$ are zero valued, they never actively contribute to the inputs of network A. Only the digit itself is *visible* (in terms of neuron activations) to the network, *i.e.* it provides forward activations $z_{ij} \neq 0$ and – according to the rules for decomposing linear mapping layers (Equation (2.31)) with ϵ-rule in Equation (2.22) or Equation (2.10) – will aggregate non-zero relevance over incoming messages $R^{(1,2)}_{i\leftarrow j}$. Correspondingly only background pixels will generate non-zero forward activations for network B and $\mathbf{x}_2$. Since the gradient of the prediction function for this particular model does not depend on an input, corresponding sensitivity maps for both models are identical regardless of input domain.

Via the construction of network B in Equation (2.63) we can see that the *mean shift compensator* ($\mathbf{mc} = -\mathbf{m}$) term can be expressed as a secondary, constant and image-shaped input to the network, to which the layer's learned weights W are applied. Since $\mathbf{mc}$ (as integrated into b_2) actively contributes non-zero forward activations (which are identical for each input image), it identifies as the source of activations (in part) responsible for the model prediction and thus absorbs a proportional quantity of relevance. Using Equations (2.63) and (2.39) we can show that

$$\begin{aligned} R^{(1,2)}_{\mathbf{x}_1,(i\leftarrow j)} &= \frac{\mathbf{x}_{1,i}w_{ij}}{z_j} \cdot R^{(2)}_j = \frac{(\mathbf{x}_{2,i} + \mathbf{mc}_i)w_{ij}}{z_j} \cdot R^{(2)}_j = \frac{\mathbf{x}_{2,i}w_{ij}}{z_j} \cdot R^{(2)}_j + \frac{\mathbf{mc}_i w_{ij}}{z_j} \cdot R^{(2)}_j \\ &\overset{\forall i:\text{Eq. (2.13)}}{\Rightarrow} R^{(1)}_{\mathbf{x}_1} = R^{(1)}_{\mathbf{x}_2} + R^{(1)}_{\mathbf{mc}}\,, \end{aligned} \tag{2.64}$$

which is illustrated in Figure 2.9 (right). Expanding the bias b_2 of network B to b and the image shaped bias input $\mathbf{mc}$ allows for an analysis of how the shift compensation reconstructs the layer activations z complementary to the inputs $\mathbf{x}_2$.

The work of (Kindermans et al., 2017a) proposes "input invariance" as an attribute for reliable explanation and interpretation of model decisions. While some methods, such as Sensitivity Analysis (Baehrens et al., 2010; Simonyan et al., 2013) fulfill this criterion, others – including LRP – do not. The explanations inherently gained from linear model predictions (see Section 2.1) – a key motivator for the decompositions performed by LRP – also do not fulfill the proposed attribute. The transparency of linear predictions as a weighting of input features however often motivates the use of this comparatively simple model, *e.g.* in human gait analysis (Phinyomark et al., 2017) and computer security (Sommer et al., 2010;

Schütt et al., 2012; Arp et al., 2014), above more complex and potent predictors in "real" application settings. Thus, an "input invariance" might not be a generally desirable attribute, especially since for linear models, an explanation as an interaction with the input can yield far more informative results than just the model gradient (see Figure 2.1 (right)). For picking a sutable method of machine learning interpretability, the semantic meaning of the explanations gained should be taken into account.

Should an invariance to modifications in the input domain as in the experiment above be desired for the interaction-based explanations provided via LRP, an application of appropriate decomposition rules (Section 2.2.6), such as the $\flat$-rule heuristic or the w^2 and z^{B} rules from (Montavon et al., 2016; Montavon et al., 2017; Montavon et al., 2018), should be considered. Figure 2.10 shows relevance maps for both networks A and B for inputs $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively and demonstrates the effect of replacing the ϵ-rule with the $\flat$-rule in the input layers at hand of a series of input samples. The application of the $\flat$-rule renders the performed relevance decomposition invariant to the transformation of the input domain between the both models. Interestingly, the relevance maps shown for the application of the $\flat$-rule to the input layers of both models are identical, since they only depend on the receptive fields of the layer's output neurons, which activate identically in both networks by construction.

2.4 LRP for Feature Extractor Pipelines and SVMs

Until recently, Bag of Words (BoW) models have been the method of choice for achieving state of the art performance in many fields, including text (Joachims, 1998; S. I. Wang et al., 2012) and image categorization (Perronnin et al., 2010; Gemert et al., 2010; Liu et al., 2011; Binder et al., 2012b; Binder et al., 2013; Sánchez et al., 2013). They are, despite the recent advances in Neural Networks, still popular and performant – especially in text analysis (X. Zhang et al., 2015; Grave et al., 2017) – and have excelled in past competitions on visual concept recognition and ranking such as Pascal VOC (Everingham et al., 2009; Everingham et al., 2010) and ImageCLEF PhotoAnnotation (Nowak et al., 2011). Often, Bag of Words feature processing pipelines are used in combination with kernelized Support Vector Machine (SVM) classifiers (Cortes et al., 1995; Müller et al., 2001; Chatfield et al., 2011). In our experience Bag of Words Models perform well for tasks with small sample sizes, whereas Deep Neural Networks are at risk to overfit due to their richer parameter structure.

In this section, the common structure of a Bag of Words feature extraction pipeline will be reiterated. After the neccessary mappings and functions for computing a forward pass for prediction are established, the LRP decomposition from Section 2.2 will be applied to the BoW model with an SVM classifier.

Lastly, an example application with different kernel functions on synthetic data demonstrates the meaningfulness of the application of LRP to a pipeline of BoW and SVM.

2.4.1 Bag of Words Models Revisited

In the following, we will consider Bag of Words features an aggregation of non-linear mappings of local features. All Bag of Words models, no matter whether based on hierarchical clustering (Moosmann et al., 2008), soft codebook mapping (Gemert et al., 2008; Gemert et al., 2010; Liu et al., 2011; Binder et al., 2013), regularized local codings (J. Yang et al., 2009; Yu et al., 2009; J. Wang et al., 2010), or Fisher vectors (Perronnin et al., 2010; Sánchez et al., 2013), share a common multi-stage procedure (see Figure 2.11):

In the first stage local features are densely computed across small regions in the image. A local feature such as SIFT (Lowe, 2004; Sande et al., 2010) is a vector computed from a region of the image, for example capturing information of interest such as shape characteristics or properties of color or texture on a local scale. In a second stage – which is performed once during training – representatives in the space of local features are computed, no matter whether they are cluster centroids obtained from k-means clustering, regions of the space as for clustering trees (Moosmann et al., 2008), or centers of distributions as for Fisher vectors (Sánchez et al., 2013). The set of representatives – in the following referred to as *visual words* – serves as a vocabulary in the context of which images can be described as vectors. In the

third stage, statistics (*e.g.* histograms) of the local features are computed relative to those visual words. These statistics are aggregated from all local features $L = \{l_j\}_{j=1\ldots N}$ within an image in order to yield a BoW representation $\mathbf{x}$, usually done by sum- or max-pooling.

The computation of statistics can be modeled by a mapping function accepting local feature vectors l as input, which are then projected into the Bag of Words feature space. Let m be such a mapping function and let $m_{(d)}$ denote the mapping onto the d-th dimension of the BoW space. We assume the very generic p-means mapping scheme for local features l as given in Equation (2.65).

$$x_{(d)} = \left(\frac{1}{|L|} \sum_{j=1}^{|L|} \left(m_{(d)}(l_j) \right)^p \right)^{\frac{1}{p}} \tag{2.65}$$

This contains sum- and max-pooling as the special cases $p = 1$ and the limit $p = \infty$.

Finally, a classifier is applied on top of these features. Our method supports the general class of classifiers based on kernel methods. For brevity we use here a kernelized SVM prediction function over BoW features x_i, training data labels y_i, kernel functions $k(\cdot,\cdot)$, and SVM model parameters b and α_i and a number S of support vectors.

$$f(x) = b + \sum_{i=1}^{S} \alpha_i y_i k(x_i, x) \tag{2.66}$$

This assumption can be extended without loss of generality to approaches using multiple kernel functions such as multiple kernel learning (Lanckriet et al., 2004; F. R. Bach et al., 2004; Kloft et al., 2009; Kloft et al., 2011; Binder et al., 2012b), structural prediction approaches with tensor product structure between features and labels as in taxonomy-based classifiers (Hwang et al., 2012; Binder et al., 2012a; Blaschko et al., 2013) or boosting-like formulations as in (Cao et al., 2009)

$$f(x) = b + \sum_{i=1}^{S} \sum_{u=1}^{K} \alpha_{i,u} k_u(x_{i(u)}, x_{(u)}). \tag{2.67}$$

Correspondingly, above extensions are covered by LRP via the merge-type Equations (2.39) and (2.38). In the following we will thus concentrate on derivations for LRP based on the prediction function in Equation (2.66).

2.4.2 LRP Decomposition for Bag of Words Models

Now, in order to apply the LRP framework to Bag of Words models, each stage (or *layer*) of the model pipeline will be considered in sequence, from the model output until the pipeline's local feature extraction stage operating on an input image. The mappings corresponding to each stage will first be expressed in terms of *forward messages* z_{ij} as in Section 2.2, from which then corresponding and backwards directed *relevance messages* $R_{i\leftarrow j}^{(l,l+1)}$ will be derived. Figure 2.11 provides an overview over the sequence of steps necessary for predicting with a Bag of Words prediction model and the analogous, backwards-directed decomposition steps.

Let us assume a fully trained Bag of Words prediction pipeline, which has made a prediction $f(\mathbf{x})$ for a given input image. For attributing relevance values $R_p^{(1)}$ to the individual pixels of the input image, LRP starts the decomposition of the predictor pipeline at the output layer, *i.e.* the prediction of the Support Vector classifier is decomposed into relevance attributions of the individual dimensions of Bag of Words representation of the image. Then, the algorithm iterates over the pipeline's mapping and transformation steps in inverse direction until the (image) input layer. Since Bag of Words prediction pipelines are in general less homogeneous in mapping structure and composition of layer types than neural network classifiers, the following paragraphs will explicitly apply the relevance decomposition rules from Equations (2.9) and (2.12) to all mapping stages from the image input to the classifier

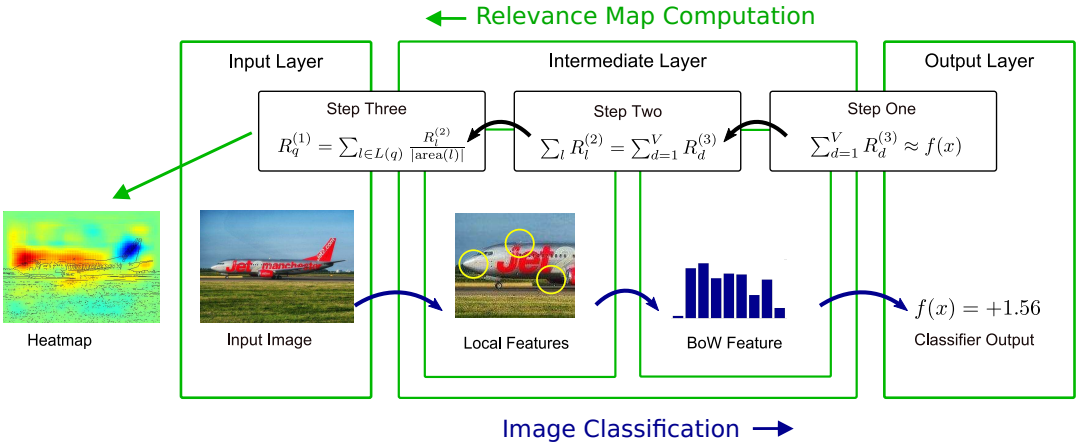

FIGURE 2.11: **Overview over the prediction forward pass through a Bag of Words model and corresponding LRP backward pass.** Local and global predictions for input images are obtained by following a series of steps through the classification- and pixel-wise decomposition pipelines. Each step taken towards the final pixel-wise decomposition has a complementing analogue within the Bag of Words classification pipeline. The calculations used during the pixel-wise decomposition process make use of information extracted by those corresponding analogues.

output. We will assign indices to the layers of data representation within the pipeline, similar to a neural network: Index (1) is assigned to the image layer of individual pixels; (2) to the image represented as a set of local descriptors; (3) to Bag of Words vector representation; and (4) to the prediction made for the image by the classifier.

Attributing Relevance to Bag of Words Dimensions for Sum-decomposable Kernel Functions

At the end of the model pipeline, a (kernel) SVM receives as input feature the output of the previous BoW mapping step and predicts whether a learned target class is present (or not) as $f(\mathbf{x})$. For brevity, we restrict our derivation to a predictor with only a single kernel function and choose to present the derivation of the kernelized prediction function in its more generally applicable dual form. Since our aim is to decompose the current prediction, we initiate the LRP algorithm with $R_f^{(4)} = f(x)$ with the aim of obtaining a relevance decomposition

$$f(x) = R_f^{(4)} \approx \sum_{d=1}^{D} R_d^{(3)} \tag{2.68}$$

for all dimensions d of the D-dimensional Bag of Words feature space.

The work of (Uijlings et al., 2012) has performed this step for the special case of one single Histogram Intersection Kernel (HIK). Such a decomposition can be generalized naturally and performed without error for all kernel functions which are sum-decomposable along input dimensions. We define a kernel function k to be sum-decomposable if there exists kernel functions $k^{(d)}$ acting on single input feature dimensions such that

$$k(x_i, x_{i'}) = \sum_{d=1}^{D} k^{(d)}(x_{i(d)}, x_{i'(d)}) \tag{2.69}$$

In this case, *e.g.* as with the linear and Histogram Intersection Kernel functions, we can fulfill Equation (2.68) with equality. Explicitly expressing $f(\mathbf{x})$ in terms of the notations introduced in Section 2.2, we define the forward propagated quantities from BoW dimensions d and the

bias b to the model output f as

$$z_{df} = \sum_{i=1}^{S} \alpha_i y_i k^{(d)}(x_{i(d)}, x_{(d)}) \text{ and } z_{bf} = b$$
$$z_f = f(\boldsymbol{x}) = \sum_d z_{df} + z_{bf} \,. \tag{2.70}$$

Then,

$$R^{(3,4)}_{d\leftarrow f} = \frac{z_{df}}{z_f} R^{(4)}_f = \frac{z_{df}}{f(\boldsymbol{x})} f(\boldsymbol{x}) = z_{df} = R^{(3)}_d$$
$$R^{(3,4)}_{b\leftarrow f} = \frac{z_{bf}}{z_f} R^{(4)}_f = \frac{z_{bf}}{f(\boldsymbol{x})} f(\boldsymbol{x}) = z_{bf} = R^{(3)}_b \tag{2.71}$$

or, more directly,

$$R^{(3)}_d = \sum_{i=1}^{S} \alpha_i y_i k^{(d)}(x_{i(d)}, x_{(d)}) \text{ and } R^{(3)}_b = b \tag{2.72}$$

Note that for Support Vector classifiers predicting two classes, the dimensionality of the output is 1, eliminating the need to sum-pool over the dimensons of f to compute $R^{(3)}_d$ in Equation (2.71). Also, as for Deep Neural Networks, we explicity attribute a quantity of the relevance at the BoW input layer to the learned bias b – exactly the amount the bias contributes to $f(\boldsymbol{x})$ as an addend – as it acts as a constant input and shifts the evaluation of the kernelized prediction by a fixed quantity. Strong equality of Equation (2.68) for sum-decomposable kernels k can be shown trivially by plugging Equations (2.71) into (2.70).

Note that in some cases, it might be desirable to distribute R_b across all input dimensions d, *e.g.* when using histograms as input features $\boldsymbol{x}$, in combination with distance-based kernel functions. A model might predict because an input does *not* have certain characteristics as expressed by the histogram features. In such cases, the *absence* of information encoded via empty histogram bins $x_d = 0$ might play an important role during inference and the majority of influence over the prediction outcome is shifted to the bias b. Corresponding to above decomposition rules, the influence of missing characteristics can not be represented by the attributed relevances $R^{(3)}_d$, yet is encoded in $R^{(3)}_b$. Hence, in such cases, we can choose to attribute the bias relevance $R^{(3)}_b$ towards the inputs as

$$\forall d: \; R^{(3)}_d = R^{(3)}_d + \frac{1}{D} R^{(3)}_b \tag{2.73}$$

or, when a meaningful projection $\boldsymbol{w}$ over all input dimensions can be obtained, as *e.g.* for a linear kernel function

$$\forall d: \; R^{(3)}_d = R^{(3)}_d + \frac{w_d}{||\boldsymbol{w}||_2} R^{(3)}_b \,. \tag{2.74}$$

Attributing Relevance to Bag of Words Dimensions for General Differentiable Kernel Functions via Taylor Decomposition

Not all kernel functions popular in machine learning are sum-decomposable into exact contributions $R^{(3)}_d$ to the prediction function $f(\boldsymbol{x})$. For the class of general differentiable kernel functions, we therefore apply Taylor Decomposition to the prediction function f as introduced in Section 2.2.5 to approximate the contribution of each input component $x_{(d)}$ linearly.

Adapting Formula (2.18) to the kernelized SVM prediction Function (2.66) yields

$$f(\boldsymbol{x}) = f(\widetilde{\boldsymbol{x}}) + \sum_{d=1}^{D} (\boldsymbol{x} - \widetilde{\boldsymbol{x}})_{(d)} \sum_{i=1}^{S} \alpha_i y_i \frac{\partial k(\boldsymbol{x}_i, \cdot)}{\partial \boldsymbol{x}_{(d)}}(\widetilde{\boldsymbol{x}}) + r. \tag{2.75}$$

Here, r is the residual term or approximation error. In above formula, the refence point (or root point) $\widetilde{\mathbf{x}}$ is the only free parameter. Our aim is to find a decomposition of $f(\mathbf{x})$ into a sum of $R_d^{(3)}$ as in Equation (2.68), which can be achieved by minimizing the error residual r and choosing a root point on the decision boundary (*i.e.* $f(\widetilde{\mathbf{x}}) = 0$) of the binary classification function:

$$R_d^{(3)} = (\mathbf{x} - \widetilde{\mathbf{x}})_{(d)} \sum_{i=1}^{S} \alpha_i y_i \frac{\partial k(\mathbf{x}_i, \cdot)}{\partial \mathbf{x}_{(d)}}(\widetilde{\mathbf{x}}) \tag{2.76}$$

By choosing a root point on the decision boundary – *i.e.* the point of reference is predicted neutrally wrt both classes – we ensure that Equation (2.68) (approximately) holds. We try to keep the residual term as small as possible by restricting the the set of root point candidates to those in close proximity to $\mathbf{x}$.

Furthermore $\widetilde{\mathbf{x}}$ provides a coordinate where f is maximally uncertain about its prediction, respectively to which the relevance decomposition is then performed. Picking a root point on the decision boundary as a point predicted neutrally retains the semantics of the relevance values computed for sum decomposable kernels, *i.e. "what makes* $\mathbf{x}$ *speak for or against the prediction target, and in what quantitiy so?"*, in contrast to an arbitrary root point, which could be another (test) sample. Picking an arbitrary $\widetilde{\mathbf{x}}$ shifts the interpretation the decomposed relevance values provide to *"what makes* $\mathbf{x}$ *more or less a member of the target class than* $\widetilde{\mathbf{x}}$*?"*, by only accounting for $f(\mathbf{x}) - f(\widetilde{\mathbf{x}})$.

Given an $\mathbf{x}$ we wish to decompose the prediction function for, we interpolate candidates for root points $\widetilde{\mathbf{x}}$ by sampling a number (≈ 50) of test samples $\mathbf{x}_s$ predicted as members of the opposing class to $\mathbf{x}$. An interval search is performed between the selected test samples and $\mathbf{x}$, until an $\widetilde{\mathbf{x}} = \gamma\mathbf{x} + (1-\gamma)\mathbf{x}_s$ with $\gamma \in [0;1]$ is found satisfying $f(\widetilde{\mathbf{x}}) = 0$. In practice, we relax the latter criterion to $|f(\widetilde{\mathbf{x}})| \leqslant \epsilon$ with a sufficiently small epsilon, *e.g.* $\epsilon = 1e{-}5$, to guarantee the convergence of the (numerically finitely precise) interval search. From all the candidate root points, we then select the one which minimizes the euclidean distance to $\mathbf{x}$ as the $\widetilde{\mathbf{x}}$ to use in Taylor approximation.

An overview of (Taylor) decompositions for in computer vision commonly used kernel functions can be found in Appendix A.1.1. Figure 2.12 intuitively describes the difference between the local derivative at the point of intrest $\mathbf{x}$ and the Taylor Decomposition wrt a root point $\widetilde{\mathbf{x}}$ and its effect on the attribution of relevance.

Computing Local Feature Relevance from Bag of Words Relevance

In the previous paragraph, we have derived relevance decompositions for sum-decomposable kernels and general differentiable kernel types. We can therefore assume that relevance attributions for Bag of Words dimensions are already given as $R_d^{(3)}$. The Bag of Words mapping stage receives as input a bag (*i.e.* multiset) of local feature descriptors $L = \{l_j\}_{j=1\ldots N}$, which are individually mapped onto one or more dimensions d of the Bag of Words Feature space and then aggregated, *e.g.* via sum-pooling. Again, we can express the very generic mapping scheme from Equation (2.65) in terms of forward messages for graph-like models. Beginning with the edge case of $p = 1$ we obtain a sum pooling feature aggregation.

$$\begin{aligned} z_{ld} &= \frac{1}{|L|} m_{(d)}(l) \\ z_d &= \sum_{l \in L} z_{ld} \end{aligned} \tag{2.77}$$

Here, z_{ld} describes contribution to the sum aggregation for each *individual* local descriptor l and Bag of Words feature dimension d and z_d consequently is the sum-pooled aggregation at output dimension d over all inputs $l \in L$, which is equal to $x_{(d)}$ in Formula (2.65). Using z_{ld} and z_d to replace their equivalents in Equation (2.10) to perform a proportional

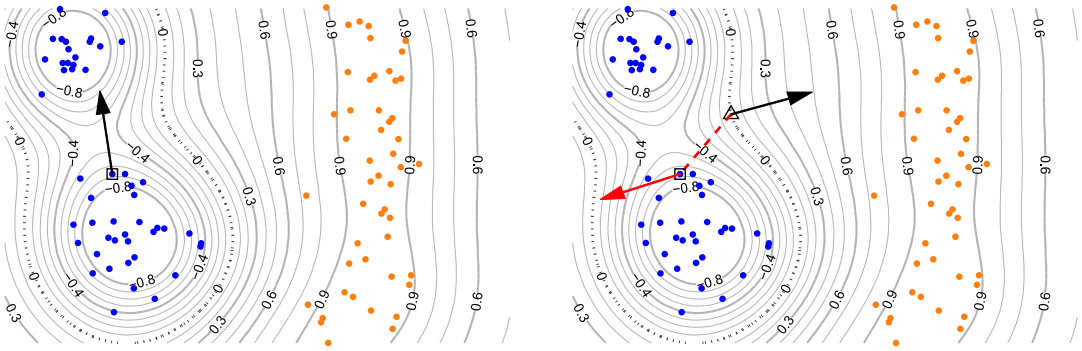

FIGURE 2.12: **Graphical comparison of local derivative at the input x against Taylor Decomposition for input x wrt a root point $\widetilde{x}$.** In both images, the sample $\mathbf{x}$ is indicated by a small box. *Left:* **(black arrow):** The local gradient of the decision function f *at* the prediction point $\mathbf{x}$, pointing towards the closest peak of the prediction function in the direction maximally changing $f(\mathbf{x})$ when changing $\mathbf{x}$. *Right:* **(black arrow):** The local gradient at the root point $\widetilde{\mathbf{x}}$ on the decision boundary. **(red, dashed):** $(\mathbf{x} - \widetilde{\mathbf{x}})$. **(red arrow, solid):** The relevance attributions R_d corresponding to Equation (2.76), visualized as a direction in $\mathbb{R}^2$. The local gradient in the left image points towards the nearest local peak in $f(\mathbf{x})$, informing about the direction of change $f(\mathbf{x})$ is most sensitive to. The toy model – an SVM with a Gaussian kernel – has been trained to perform a binary prediction thresholded at 0 on both class densities. The relevance decomposition wrt to $\widetilde{\mathbf{x}}$ on the decision hyperplane attributes dominantly large negative relevance to the horizontal direction. This *global* relevance attribution tells us that the prediction point $\mathbf{x}$ is mainly predicted as the blue class (labelled negatively) due to its horizontal position.

decomposition of relevance yields

$$R^{(2,3)}_{l\leftarrow d} = \frac{z_{ld}}{z_d} R^{(3)}_d = \frac{m_{(d)}(l)}{\sum_{l'\in L} m_{(d)}(l')} R^{(3)}_d \,. \tag{2.78}$$

Note that there might be dimensions d in Bag of Words feature space which are not reached by any mapping z_{ld} for the current image, *i.e.* $\exists d \forall l : z_{ld} = 0$. The top layer relevance $R^{(3)}_d$ might still be greater or smaller than zero. This case is especially likely to happen when a combination of very sparse histogram mappings such as Vector Quantization (Cosman et al., 1993) is used in combination with distance-based kernel embeddings, for example the Gaussian RBF kernel (Steinwart et al., 2006). Here, the $R^{(3)}_d$ attributed to dimensions without local feature mapping contributions corresponds to the model using the *absence of image features* to predict for or against the target class. We extend above decomposition Formula (2.78) to

$$R^{(2,3)}_{l\leftarrow d} = \begin{cases} \frac{z_{ld}}{z_d} R^{(3)}_d & ; \exists l' : z_{l'd} \neq 0 \\ \frac{1}{|L|} R^{(3)}_d & ; \forall l' : z_{l'd} = 0 \end{cases} . \tag{2.79}$$

As before, for Bag of Words dimensions d which are reached by mappings from any local feature input, the relevance quantity $R^{(3)}_d$ is distributed proportionally to all local descriptors mapping to dimension d according to the quantity of their contribution. The relevance scores attributed to Bag of Words dimension *without* local feature mappings are distributed uniformly across all local descriptors $l \in L$. The relevance score for each local descriptor l is then computed by sum-pooling all incoming relevance messages.

$$R^{(2)}_l = \sum_{d=1}^{D} R^{(2,3)}_{l\leftarrow d} \tag{2.80}$$

A decomposition performed as in Equation (2.79) avoids division by zero and conserves the total amount of relevance between layers, *i.e.* $\sum_l R_l^{(2)} = \sum_d R_d^{(3)}$.

We like to point out that this property holds also in the case when mappings $m_{(d)}$ can become negative as a consequence of the definition used in Equation (2.79). For that reason our approach is also applicable to Fisher vectors (Perronnin et al., 2010) (see Chapter 4) and regularized coding approaches (J. Yang et al., 2009; Yu et al., 2009; J. Wang et al., 2010). Furthermore note that Equation (2.79) has no explicit dependence on the way how the local features are pooled in the mapping Formula (2.65) and this might be inappropriate weighting for max-pooling or general p-means pooling.

We can extend this definition to reflect the usage of p-means pooling

$$M_p(x_1, \ldots, x_n) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i^p \right)^{\frac{1}{p}}, \tag{2.81}$$

which may yield different results in prediction and local decomposition than sum pooling. The extension is well-defined for non-negative mappings $m_{(d)} \geqslant 0$ and any value of p and for arbitrary mappings when combined with a value of p from the natural numbers. We extend Equations (2.77) and (2.78) to

$$\begin{aligned} z_{ld} &= \frac{1}{|L|} (m_{(d)}(l))^p \\ \frac{z_{ld}}{z_d} R_d^{(3)} &= \frac{\left(m_{(d)}(l)\right)^p}{\sum_{l' \in L} \left(m_{(d)}(l')\right)^p} R_d^{(3)} \end{aligned} \tag{2.82}$$

which can be plugged into Equation (2.79). The quotient in Equation (2.82) converges to an indicator function for the maximal mapping element in the limit $p \to \infty$ which is consistent to max-pooling:

$$\frac{\left(m_{(d)}(l)\right)^p}{\sum_{l' \in L} \left(m_{(d)}(l')\right)^p} \to \mathbb{I}_{\{\text{argmax}_{l'}(m_{(d)}(l'))\}}(l) \tag{2.83}$$

Computing Pixel Relevance from Local Feature Relevance

The input layer of the model usually are the pixels of an input image. To compute relevance scores $R_p^{(1)}$ for individual pixels p, we use information about the placement and geometry of all local feature descriptors l for which $R_l^{(2)}$ have been computed in the previous step. In terms of Layer-wise Relevance Propagation a local feature is a computation unit which has as many inputs as the number of pixels it is covering. In general, the contribution of a single pixel to a local descriptor is not clearly defined, for example when descriptors collecting quantiles about color information are used. Therefore, without assumption of any further structure we distribute the relevance of the local feature equally to all its covered pixels. We define a function area(l), which returns all pixel coordinates included in the computation of a given descriptor l, and a function $L(p) = \{l | p \in \text{area}(l)\}$, returning all local descriptors using a pixel p as input. Assuming equal contribution for all pixels $p \in \text{area}(l)$ yields

$$\begin{aligned} z_{pl} &= 1 \\ z_l &= \sum_{p \in \text{area}(l)} z_{pl} \end{aligned} \tag{2.84}$$

and

$$\begin{aligned} R_{p \leftarrow l}^{(1,2)} &= \frac{z_{pl}}{z_l} R_l^{(2)} = \frac{R_l^{(2)}}{|\text{area}(l)|} \\ R_p^{(1)} &= \sum_{l \in L(p)} R_{p \leftarrow l}^{(1,2)} \end{aligned} \tag{2.85}$$

This approach is functionally identical to the $\flat$-rule from Equations (2.27) and (5.1). For relevance visualization, we implement the color mapping procedure described in Section 2.3.3.

Algorithm 2 in Appendix A.1 provides an algorithmic view of LRP for prediction pipelines comprised of BoW mappings and kernel SVM classifiers.

2.4.3 Example Application on Toy Data with Different Kernels

Let us provide a qualitative comparison for two models trained to solve a simple image recognition task on toy data. The results show that relevance decomposition yields plausible results for both sum-decomposable kernels and general differentiable kernel functions.

As data, we generate large grayscale images showing regular polygons – from tetragons to heptagons – and round objects with different, ℓ_p-based shapes. Polygons and round shapes are scattered uniformly across the generated images without overlap. From those large images, we crop non-overlapping (102×102) pixel sized sample images for training. A sample image is labelled positively ($Y = 1$) if the center of mass of at least one polygon is within the image. Otherwise, $Y = -1$. For testing, we generate another batch of images, from which we crop overlapping (102×102) pixel large images at 34 pixel intervals.

For each sample, we compute SIFT (Lowe, 2004) descriptors for fixed scale parameters $\sigma \in \{2.0, 3.0\}$ from a grid of keypoint locations 6 and 9 pixels apart respectively. We rotate the SIFT descriptor mask relative to the main gradient over pixel intensities. Then, for each local descriptor scale σ separately, we use k-means ($k = 128$) clustering (MacQueen, 1967) over all training set local descriptors in order to compute a visual vocabulary for Bag of Words feature space mapping. For the computation of Bag of Words vectors, we use a sum-pooled rank-mapping approach (Binder et al., 2013)

$$x_{(d)} = \frac{1}{|L|} \sum_{j=1}^{|L|} m_{(d)}(l_j)$$
$$m_{(d)}(l) = \begin{cases} p^{-\mathrm{rk}_d(l)} & \text{if } \mathrm{rk}_d(l) \leqslant n \\ 0 & \text{else} \end{cases}, \tag{2.86}$$

softly relating a local descriptor l to the closest n k-means cluster centroids. Following (Binder et al., 2013), we set $p = 2$ and $n = 4$. Here $\mathrm{rk}_d(l)$ is a function describing the position of the d-th k-means cluster centroid in an ascendingly sorted ranking of euclidean distances between l and all cluster centroids. For each SIFT scale parameter σ we thus obtain one 128-dimensional Bag of Words vector representation per sample image, to which ℓ_1-normalization is applied.

To compare relevance decompositions for sum-decomposable and differentiable kernel functions k in an input scale invariant manner, we use multiple kernel learning (Kloft et al., 2009; Kloft et al., 2011) (one kernel k per scale σ) to train Support Vector predictors as in Equation (2.67), using linear optimization for the kernel mixture weights and quadratic optimization for the model parameters of the SVM alternatingly via the SHOGUN machine learning toolbox (Sonnenburg et al., 2010).

For the first model we select Histogram Intersection kernels (Swain et al., 1991; Maji et al., 2008) (Equation (A.2)) to represent the class of sum-decomposable kernel functions. The second SVM model is based on the differentiable χ^2 kernel embeddings (Equation (A.6)). Both kernel functions perform non-linear mappings from input space into feature space and both models share the same Bag of Words preprocessing pipeline. Both trained SVM predictors perform reasonably well in patch-wise prediction ($\approx 97\%$ accuracy) on the hold out test data, with the model based on χ^2 kernel mappings slightly outperforming (+1.6%) the competing model.

We apply LRP to predictions made by both models on the test image patches. Since test image patches are extracted from their respective source images with overlap, we first merge the pixel-wise relevance scores obtained for each patch at the original location of the patch by locally averaging overlapping relevance maps. After merging all relevance patches into a single, larger relevance map, color space mapping is applied as described in Section 2.3.3.

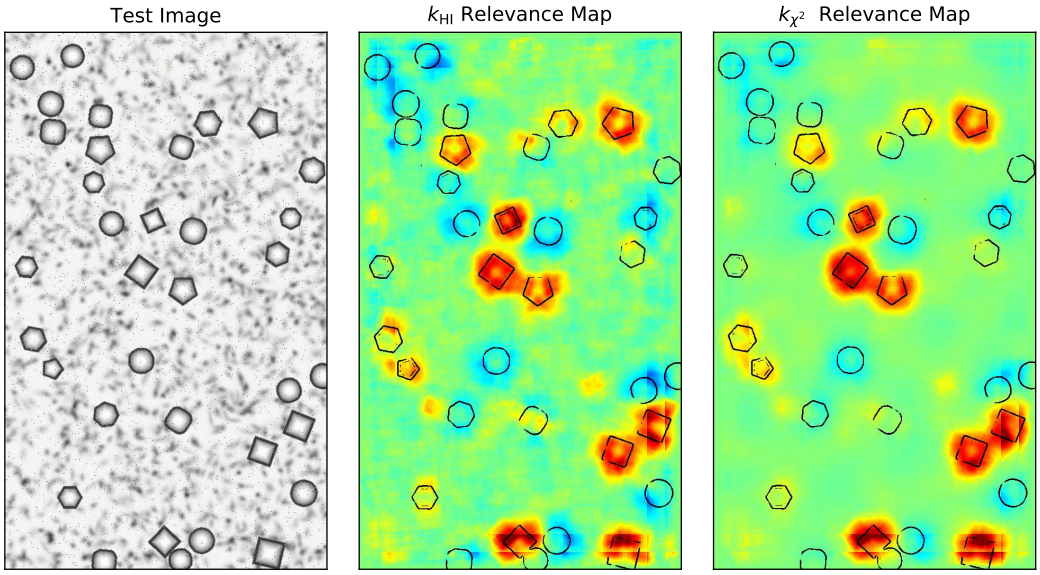

FIGURE 2.13: **Relevance maps for Bag of Words features and different kernel types.** *Left:* The input images, from which smaller patches are extracted as test images. *Middle and Right:* Reassembled relevance maps for the full test image from individual test patch relevance maps for a model based on Histogram Intersection kernels and χ^2 kernels. Relevance decompositions were computed on tiles of size 102×102 and having a regular offset of 34 pixels.

The result is a single large heatmap visualizaton of the attributed relevance scores per model, as shown in Figure 2.13.

Figure 2.13 shows that the first model using the sum-decomposable kernel function yields understandable and expected relevance decomposition results: Pixels showing polygons receive positive relevance values and image areas covered with round shapes are attributed negative relevance scores, but less pronounced than the other class shapes' positive relevance responses.

That result is consistent with the labelling paradigm used in training. Since image patches are labelled positively once a polygon-based shape is sufficiently located within the image borders – regardless of the presence of round shapes – polygon-based shapes are a much stronger indicator for class membership than round shapes. Correspondingly, due to the task of *detecting* polygons, positive evidence on the target shapes is much more pronounced than on the round samples describing the negative class. Interestingly, polygon shapes with a higher amount of vertices tend to receive lower (or even negative) amounts of relevance due to their similarity to round shapes. The generated background pattern does in general not affect the model in its decision and thus receives relevance scores close to zero. When patterns in the image background align to form edge and corner-like structures, however (as in Figure 2.13, both relevance maps, at $\approx \frac{1}{3}$ of the height of the images), positive relevance is attributed to those structures, speaking for non-target features affecting the model prediction.

We remark that the shapes are placed at arbitrary locations in the image tiles of the training set, which have not been encoded in any feature mapping stage. Thus, the relevance decomposition of the models' respective prediction functions $f(\mathbf{x})$ is able to mark structures on a smaller scale than the classifier was trained on, although position information was never provided during training.

We wish to point out that relevance – be that positive or negative – never exactly fits the extent of a shape in above depiction. The granularity of the relevance attributions is limited by the scale of the SIFT features used. Here, a SIFT feature of $\sigma = 3.0$ covers a square area in an input image patch with a side length of $4 \cdot 3 \cdot 3 = 36$ pixels. Since a SIFT feature is a 128-dimensional vector of histograms compiled from the feature's *receptive field*, we uniformly distribute the relevance score attributed to that feature uniformly over the covered pixels. This necessary heuristic is functionally equivalent to the application of

the $\flat$-rule from Equations (2.27) and (5.1). Finally we remark that the rank-mapping is a discontinuous (and thus indifferentiable) weighting scheme for BoW feature dimensions, yet the LRP yields reasonable explanations.

The (approximated) relevance decompositions computed for the second model using a χ^2-kernel function are qualitatively similar to the decompositions gained from the first, sum-decomposable model. Relevance allocation on background image areas is, in comparison, smoother and less discontinuous. This can be explained at hand of the decomposition rule derived for SVM predictors with χ^2-kernels in Equation (A.6), within which the prediction function $f(\mathbf{x}) = \sum_{i=1}^{S} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i)$ itself (without the bias) is acting as a scaling factor to the approximate linearization: Given a sample $\mathbf{x}$ predicted uncertainly with $f(\mathbf{x}) \approx 0$, relevances attributed to Bag of Words dimensions will also be small. Test samples yielding more pronounced predictions in either direction will thus result in relevance attributions of less dampened amplitude. However, this also means that tiles showing only partly covered shapes may affect relevance values on those shapes fully shown in neighboring tiles after aggregating all smaller tile-based relevance maps into a single larger one.

2.5 Limitations

Layer-wise Relevance Propagation has so far been designed – as described in this chapter – with feed-forward type models in mind, using sum-pooling, max-pooling or generalized p-means pooling for aggregating forward mappings z_{ij}. In (S. Bach, 2013) early intuitions for decomposing product-based feature aggregations are discussed, *e.g.* as they occur in (conditional) probabilistic models such as Markov chains. However, this has not been explored yet extensively and is subject to future work.

The implicit introduction and use of the $\flat$-rule for decomposing the local feature extraction layer of Bag of Words methods points out the necessity of knowledge about the performed mappings z_{ij}, in order to meaningfully compute a relevance backward pass. At this point, the $\flat$-rule constitutes a heuristic workaround, which can not fully represent the operations performed during the computation of local descriptors.

For Neural Network type models, we have shown in Section 2.3.2 that the LRP backward pass can efficiently be implemented as a modified gradient backward pass. For indifferentiable mappings, *e.g.* as part of a Bag of Words pipeline, such a convenient solution might not exist. Especially in case the forward mappings z_{ij} are not sparse and the number of input features i and output features j is high, the computation of relevance decompositions might become inefficient: Either one has to (re)compute the (number of inputs $\times$ number of outputs) mappings z_{ij} and z_j for each decomposition backward step explicitly – for example due to memory constraints – or, given enough system memory, may remember the z_{ij} during the forward pass through the model. In either case, the choice to be made represents a trade-off between run-time efficiency and memory efficiency.

Chapter 3

Evaluating and Understanding Heatmaps

In Chapter 2 we have introduced the method of Layer-wise Relevance Propagation and have demonstrated the meaningfulness of its application to Bag of Words and DNN type models. However, several approaches have recently been proposed enabling one to understand and interpret the reasoning embodied in a DNN for a single test image. These methods quantify in a way the *importance* of individual pixels – wrt the classification decision and the semantic context of the approach – and allow a visualization in terms of a heatmap in pixel/input space. While the usefulness of heatmaps can be judged subjectively by a human, human expectation might be sidetracked, *e.g.* by visual heatmap aesthetics, and not necessarily reflect how important inputs are to the model: An objective quality measure is missing.

In this chapter we present a general methodology based on input (region) perturbation for evaluating ordered collections of pixels such as heatmaps. We compare heatmaps computed by three different methods on the SUN397, ILSVRC2012 and MIT Places data sets. Our main result is that Layer-wise Relevance Propagation qualitatively and quantitatively provides a better explanation of what made a DNN arrive at a particular classification decision than the sensitivity-based approach or the Deconvolution method. We provide theoretical arguments to explain this result and discuss its practical implications. Finally, we investigate the use of heatmaps for unsupervised assessment of Neural Network performance. This chapter covers (Samek et al., 2017) and in part (S. Bach et al., 2015).

3.1 Introduction

Only recently, the transparency problem has been receiving more attention for general nonlinear estimators (Braun et al., 2008; Zien et al., 2009; Baehrens et al., 2010; Hansen et al., 2011; Montavon et al., 2013a; Vidovic et al., 2015). Namely, the difficulty to intuitively and quantitatively understand the result of model inference, *i.e.* , for an *individual* novel input data point, *what* made the trained DNN model arrive at a particular response. Note that this aspect differs from feature selection (Guyon et al., 2003), where the question is: Which features are on average salient for the *ensemble* of training data. Several methods have been developed to understand what a DNN has learned (Erhan et al., 2010; Montavon et al., 2011). While a large body of work is dedicated to visualize particular neurons or neuron layers in DNNs (Krizhevsky et al., 2012; Szegedy et al., 2013; Goodfellow et al., 2014; Zeiler et al., 2014; Mahendran et al., 2015; Yosinski et al., 2015; Dosovitskiy et al., 2016), we focus here on methods which visualize the impact of particular regions of a given and fixed single image for a prediction of this image. (Zeiler et al., 2014) have proposed in their work a network propagation technique to identify patterns in a given input image that are linked to a particular DNN prediction, *i.e.* to visualize which (input) patterns are responsible for maximizing the activation of certain output neurons. This method runs a backward algorithm that reuses the weights at each layer to propagate the prediction from the output down to the input layer, leading to the creation of meaningful patterns in input space, informing about *"which patterns in input space maximize this network output"*. This approach was designed for a particular type of Neural Network, namely Convolutional nets with max-pooling and rectified linear units. A limitation of the Deconvolution method is the absence of a particular theoretical criterion that would directly connect the predicted output to the produced pattern in

a quantifiable way. Furthermore, the usage of image-specific information for generating the backprojections in this method is limited to max-pooling layers alone.

Further previous work has focused on understanding non-linear learning methods such as DNNs or kernel methods (Baehrens et al., 2010; Rasmussen et al., 2012; Simonyan et al., 2013) essentially by Sensitivity Analysis in the sense of scores based on partial derivatives at the given sample. Partial derivatives look at local sensitivities detached from the decision boundary of the classifier. (Simonyan et al., 2013) applied partial derivatives for visualizing input sensitivities in images classified by a deep neural network. Note that although (Simonyan et al., 2013) describes a Taylor series, it relies on partial derivatives at the given image for computation of results. In a strict sense partial derivatives do not explain a classifier's decision (*"what speaks for the presence of a car in the image"*), but rather tell us *"what change would make the image more or less belong to the category car"* (also, see Figure 2.12 in Section 2.4.2). As shown later these two types of explanations lead to very different results in practice.

Layer-wise Relevance Propagation aims at explaining the difference of a prediction $f(\mathbf{x})$ relative to the neutral state $f(\mathbf{x}) = 0$ and relies on a conservation and proportionality principle to propagate the prediction back without using gradients (see Chapter 2, Equation (2.10)). This principle ensures that the network output activity is fully redistributed through the layers of a DNN onto the input variables, *i.e.* neither positive nor negative evidence is lost.

In the remainder of this chapter we will denote the visualizations produced by the above methods as *heatmaps*. While per se a heatmap is an interesting and intuitive tool that can already allow to achieve transparency, it is difficult to *quantitatively* evaluate the quality of a heatmap. In other words we may ask: What exactly makes a "good" heatmap? A human may be able to intuitively assess the quality of a heatmap, *e.g.* , by matching with a prior of what is regarded as being relevant (see Figure 3.1). However, a human-designed prior might not match up with parts of a heatmap which make it good wrt to the evaluated combinatinon of input and model. For practical applications, an automated objective and quantitative measure for assessing heatmap quality becomes necessary. Note that the validation of heatmap quality is important if we want to use it as input for further analysis. For example we could run computationally more expensive algorithms only on relevant regions in the image, where relevance is detected by a heatmap.

In this chapter we contribute by
(1) pointing to the issue of how to objectively evaluate the quality of heatmaps. To the best of our knowledge this question has not been raised so far. Then,
(2) we introduce a generic framework based on input perturbation for evaluating heatmaps, from binary inputs to color images. By
(3) comparing three different heatmap computation methods on three large datasets, we note that the relevance-based LRP algorithm is more suitable for explaining the classification decisions of DNNs than the sensitivity-based approach (Simonyan et al., 2013) and the Deconvolution method (Zeiler et al., 2014). Lastly,
(4) we investigate the use of relevance maps for the assessment of Neural Network performance.

The next section briefly introduces three existing methods for computing heatmaps. Section 3.4 discusses the heatmap evaluation problem and presents a generic framework for this task. Two experimental results are presented in Section 3.5: The first experiment compares different heatmapping algorithms on SUN397 (Xiao et al., 2010), ILSVRC2012 (Russakovsky et al., 2015) and MIT Places (Zhou et al., 2014b) datasets and the second experiment investigates the correlation between heatmap quality and neural network performance on the CIFAR-10 data set (Krizhevsky, 2009). We conclude the chapter in Section 3.7 and give an outlook.

3.2 Understanding Properties of Methods for Explaining Neural Networks

In the following we focus on images, but the presented techniques are applicable to any type of input domain whose elements can be processed by a Neural Network. Let us consider an

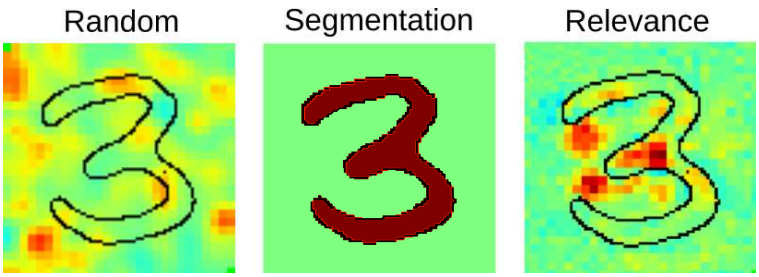

FIGURE 3.1: **Comparison of three exemplary heatmaps for the image of a "3"**. *Left*: The randomly generated heatmap lacks interpretable information. *Middle*: The segmentation heatmap (binary values) focuses on the whole digit without indicating what parts of the image were particularly relevant for classification. Since it does not suffice to consider only the highlighted pixels for distinguishing an image of a "3" from images of an "8" or a "9", this heatmap is not useful for explaining classification decisions. *Right*: A relevance heatmap indicates which parts of the image are used by the classifier. Here the heatmap reflects human intuition very well because the horizontal bar in the vertical center together with the strokes on the left are strong evidence that the image depicts a "3" and not any other digit.

image $\mathbf{x} \in \mathbb{R}^d$, decomposable as a set of pixel values $\mathbf{x} = \{x_p\}$ where p denotes a particular pixel, and a classification function $f : \mathbb{R}^d \to \mathbb{R}$. The function value $f(\mathbf{x})$ can be interpreted as a score indicating the certainty of the presence of a certain type of object(s) in the image. Such functions can be learned very well by a Deep Neural Network. Throughout the chapter we assume Neural Networks to consist of multiple layers of neurons and reuse the notation for (linear) forward transformations (Equation (2.29)) and neuron activations (Equation (2.30)) introduced in Section 2.3.1.

A *heatmap* $\mathbf{h} = \{h_p\}$ assigns each pixel p a value $h_p = \mathcal{H}(\mathbf{x}, f, p)$ according to some function $\mathcal{H}$, typically derived from a class discriminant f. Since $\mathbf{h}$ has the same dimensionality as $\mathbf{x}$, it can be visualized as an image. In the following we review three recent and prototypical methods for computing heatmaps in detail, all of which perform a backward propagation pass on the Network:
(1) A Sensitivity Analysis based on Neural Network partial derivatives,
(2) the so-called Deconvolution method and
(3) the Layer-wise Relevance Propagation algorithm.
Figure 3.2 briefly summarizes the methods. In the (very) recent past, many further methods for interpreting (specific) DNNs have emerged, in part extending or modifying the methods listed above. Section 3.2.4 provides a concise overview about these methods.

3.2.1 Sensitivity Heatmaps

A well-known tool for interpreting non-linear classifiers is Sensitivity Analysis (Baehrens et al., 2010). It was used in (Simonyan et al., 2013) to compute saliency maps of images classified by Neural Networks. In this approach the sensitivity of a pixel h_p is computed by using the norm $\|\cdot\|_{\ell_q}$ over partial derivatives ((Simonyan et al., 2013) used $q = \infty$) for the color channel c of a pixel p:

$$h_p = \left\| \left(\frac{\partial}{\partial x_{p,c}} f(\mathbf{x}) \right)_{c \in (r,g,b)} \right\|_{\ell_q} \tag{3.1}$$

Partial derivatives are obtained efficiently by running the backpropagation algorithm (Rumelhart et al., 1986) throughout the multiple layers of the Network. The backpropagation rule from one layer to another layer, where $x^{(l)}$ and $x^{(l+1)}$ denote the neuron activities at two

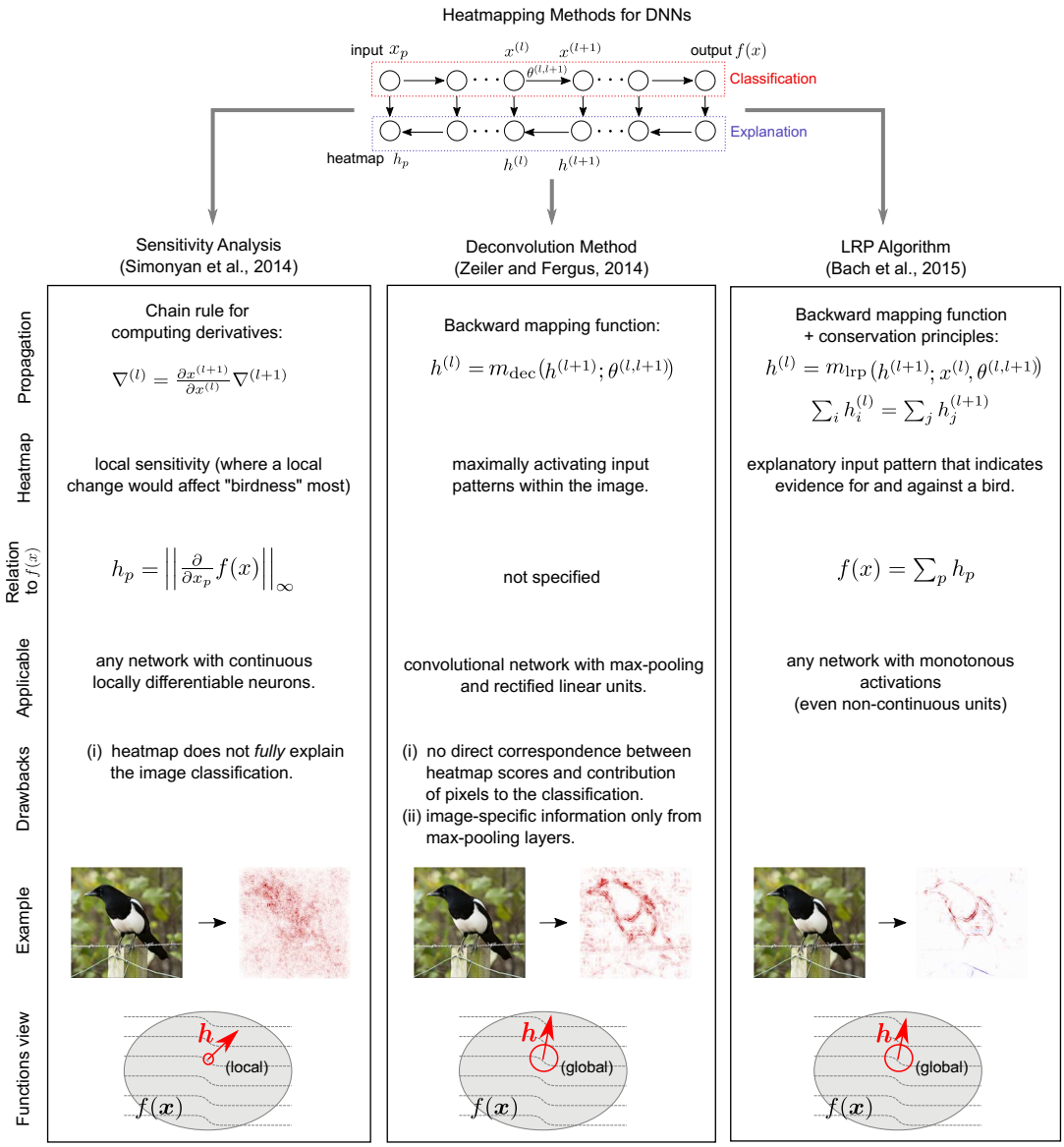

FIGURE 3.2: **Comparison of the three heatmap computation methods used in this chapter.** *Left:* Sensitivity heatmaps are based on partial derivatives, *i.e.* measure which pixels, when changed, would make the image belong less or more to a category (local explanations). The method is applicable to generic architectures with differentiable units. *Middle:* The Deconvolution method applies a Convolutional Network g to the output of another Convolutional Network f. Network g is constructed in a way to "undo" the operations performed by f. Since negative evidence is discarded and scores are not normalized during the backpropagation, the relation between heatmap scores and the classification output $f(\mathbf{x})$ is unclear. *Right:* LRP exactly decomposes the classification output $f(\mathbf{x})$ into pixel relevances by observing the layer-wise conservation principle, *i.e.* evidence for or against a category is not lost. The algorithm does not use gradients and is therefore applicable to generic architectures (including nets with non-continuous units). LRP globally explains the classification decision and heatmap scores have a clear interpretation as evidence for or against a category.

consecutive layers is given by:

$$\frac{\partial f}{\partial x^{(l)}} = \frac{\partial x^{(l+1)}}{\partial x^{(l)}} \frac{\partial f}{\partial x^{(l+1)}} \tag{3.2}$$

The backpropagation algorithm performs the following operations in the various layers and obtains heatmaps containing the following information:

Unpooling: The gradient signal is redirected onto the input neuron(s) to which the corresponding output neuron is sensitive. In the case of max-pooling, the input neuron in question is the one with maximum activation value.

Nonlinearity: Denoting by $z_i^{(l)}$ the preactivation of the ith neuron of the lth layer, backpropagating the signal through a rectified linear unit (ReLU) defined by the map $z_i^{(l)} \to \max(0, z_i^{(l)})$ corresponds to multiplying the backpropagated gradient signal by the step function $1_{\{z_i^{(l)}>0\}}$. The multiplication of the signal by the step function makes the backward mapping discontinuous, and consequently strongly local.

Filtering: The gradient signal is convolved by a transposed version of the convolutional filter used in the forward pass.

Heatmap interpretation: Sensitivity Analysis measures how much small changes in the pixel value locally affect the network output. Large values of h_p denote pixels which largely affect the classification function f if changed. Note that the direction of change (*i.e.* the sign of the partial derivative per color channel) is lost when using the norm.

In the experiments, we compute heatmaps by using Equation (3.1) with the norms $q = \{2, \infty\}$.

3.2.2 Deconvolution Heatmaps

Another method for heatmap computation was proposed in (Zeiler et al., 2014) and uses a process termed Deconvolution. Similarly to the backpropagation method to compute the function's gradient, the idea of the Deconvolution approach is to map the activations from the network's output back to pixel space using a backpropagation rule

$$R^{(l)} = m_{\text{dec}}(R^{(l+1)}; \theta^{(l,l+1)}). \tag{3.3}$$

Here, $R^{(l)}, R^{(l+1)}$ denote the backward signal as it is backpropagated from one layer to the previous layer, m_{dec} is a predefined function that may be different for each layer and $\theta^{(l,l+1)}$ is the set of parameters connecting two layers of neurons. This method was designed for Convolutional Networks with max-pooling and rectified linear units, but it could also be adapted in principle for other types of architectures. The following set of rules is applied to compute Deconvolution heatmaps.

Unpooling: The locations of the maxima within each pooling region are recorded and these recordings are used to place the relevance signal from the layer above into the appropriate locations. For Deconvolution this seems to be the only place besides the classifier output where image information from the forward pass is used, in order to arrive at an image-specific explanation.

Nonlinearity: The relevance signal at a ReLU layer is passed through a ReLU function during the Deconvolution process.

Filtering: In a convolution layer, the transposed versions of the trained filters are used to backpropagate the relevance signal. This projection does not depend on the neuron activations $x^{(l)}$.

The unpooling and filtering rules are the same as those derived from gradient propagation (*i.e.* those used in Section 3.2.1). The propagation rule for the ReLU nonlinearity differs from backpropagation: Here, the backpropagated signal is not multiplied by a discontinuous step function, but is instead passed through a rectification function similar to the one used in the forward pass. Note that unlike the indicator function, the rectification function is continuous.

Heatmap interpretation: Deconvolution heatmaps identify meaningful patterns in input space, which maximally (among other patterns) activate the analyzed network output.

For Deconvolution we apply the same color channel pooling methods (2-norm, ∞-norm) as for Sensitivity Analysis.

3.2.3 Relevance Heatmaps

Layer-wise Relevance Propagation is derived from two principles (see Section 2.2): A *proportionality* principle (see Equation (2.13)) dictates the distribution of the propagated quantity towards the model inputs to be in proportion to the inputs' contribution to the activation of respective successor neurons. A layer-wise *conservation* principle (see Equation (2.5)), which forces the propagated quantity (*e.g.* , evidence for a predicted class) to be preserved between neurons of two adjacent layers. The heatmap resulting from LRP satisfies $\sum_p h_p = f(\mathbf{x})$ (where $h_p = R_p^{(1)}$) and is said to be consistent with the evidence for the predicted class. Stricter definitions of conservation that involve only subsets of neurons can further impose that relevance is locally redistributed in the lower layers. The propagation rules for each type of layer and interpretation of resulting heatmaps are given below:

Unpooling: Like for the previous approaches, the backward signal is redirected proportionally onto the location for which the activation was recorded in the forward pass.

Nonlinearity: The backward signal is simply propagated onto the lower layer, ignoring the rectification (or in general component-wise activation functions) operation. Note that this propagation rule satisfies both Equations (2.5) and (2.13).

Filtering: The backward signal is computed via decomposition of the forward signal, *i.e.* proportional to weighted contributions in the forward pass. Chapter 2 proposed the ϵ-rule (Equation (2.22)) and the $\alpha\beta$-rule (Equation (2.26)), among others.

Heatmap interpretation: Due to the conservation and proportionality principles, relevance heatmaps show which input pixels contribute how much, towards ($h_p > 0$), against ($h_p < 0$) or not at all ($h_p \approx 0$) to the analyzed model output, given an input $\mathbf{x}$.

In the experiments section we use for consistency the same settings as in (S. Bach et al., 2015) without having optimized the parameters, namely the LRP variant from Equation (2.26) with $\alpha = 2$ and $\beta = -1$ (which will be denoted as LRP in subsequent figures), and twice LRP from Equation (2.22) with $\epsilon = 0.01$ and $\epsilon = 100$. An implementation of LRP is described in (Lapuschkin et al., 2016b)[1].

3.2.4 Overview over Further Interpretability Methods and Related Work

In the past view years, many methods for interpreting DNN classifiers have been proposed in literature. The work of (Kindermans et al., 2017b) categorizes these interpretability methods into three groups, namely *Function*-based methods, *Signal*-based methods and *Attribution* methods. In the remainder of this section, we add another category – that of *Sampling*- or *Learning*-based methods – and provide a brief overview over recent approaches to interpretability while drawing connections to LRP where applicable.

Function-based Interpretation

Function-based methods explain the function represented by the trained predictor f in input space. Due to the highly non-linear nature of DNN models, function-based explanations are in many cases only approximate. (Kindermans et al., 2017b) groups the gradient of a predictor function f and Sensitivity Analysis (Simonyan et al., 2013) (SA, see above) into this category.

Signal-based Interpretation

Signal-based methods such as the Deconvolution (Zeiler et al., 2014) method aim to visualize the signal $\mathbf{s}$[2] in input space which caused the neural network decision (Zeiler et al., 2014).

Understanding that Deconvolution does not work well in networks without max-pooling layers, (Springenberg et al., 2015) introduces *Guided Backpropagation* (GB) as an extension to the Deconvolution method. GB can be applied to a wider range of network architectures,

[1] The toolbox is downloadable from http://heatmapping.org and https://github.com/sebastian-lapuschkin/lrp_toolbox.

[2] The signal $\mathbf{s}$ as part of the input data $\mathbf{x}$ next to distracting noise patterns $\mathbf{d} : \mathbf{x} = \mathbf{s} + \mathbf{d}$ (Kindermans et al., 2017b)

e.g. the all-convolutional network proposed in (Springenberg et al., 2015). In contrast to Deconvolution, which applies the ReLU function when backward-passing through the non-linearity, GB also masks the ReLU-processed backward gradient with a ReLU of the forward-signal (*i.e.* the inputs of a layer). This propagation approach prevents the propagation of negative signals altogether and makes the presence of max-pooling switches non-mandatory for obtaining image-specific analyses.

PatternNet (Kindermans et al., 2017b) (PN) is a learning-based approach based on the assumption that the prediction function f does not only process the signal part of $\mathbf{x}$, but also cancels out distracting noise patterns present in the input. PN trains a reusable *signal estimator* on input data once, which allows the method to provide a layer-wise back-projection of the estimated signal component $\mathbf{s}$ of $\mathbf{x}$ in input space. This approach is similar to the visualization of activation patterns common in neuro-imaging (Haufe et al., 2014). Since PN involves a (one time) learning step, it can also be categorized into below class of learning-based explaining methods.

Attribution or Interaction-based Interpretation

Attribution-based approaches combine function-based and signal-based approaches and aim to explain how a model reacts in its decision making to a given input. Like LRP and DTD, the following methods try to compute by how much the components of a given input contribute (differentially) to $f(\mathbf{x})$. Several later contributions in this category implement special cases of LRP. We provide corresponding remarks to illustrate the similarities and distinctions.

Class Activation Mapping (Zhou et al., 2016) (CAM) can be performed for DNNs following a particular architecture of several convolutional layers, followed by Global Average Pooling (Lin et al., 2014) (GAP) and terminated with a dense layer for classification. A CAM can then be computed by combining the GAP and the final dense layer into a single operation, and only computing the sum over the channel-wise weighted inputs to the GAP, for an output class of choice. The resulting CAM then has the same spatial extent as the input to the GAP with only one feature channel remaining. This approach is equivalent to the application the naive LRP decomposition rule (*i.e.* the ϵ-rule with $\epsilon = 0$) to the final dense layer and then aggregating relevance over the channel axis of the tensor, creating a very coarse localization map.

Gradient-weighted Class Activation Mapping (Selvaraju et al., 2017) (Grad-CAM) is a strict generalization of CAM and does not necessarily require the presence of a GAP. Grad-CAM computes coarse localization maps by computing partial derivatives for a desired class output wrt all feature maps of a (rectified) convolution layer of interest within the CNN-part of the network. The activation maps are then the result of a rectified (to only keep positive responses) multiplication of the convolution layer's feature map and the globally average pooled partial derivatives (*i.e.* a weight per feature map). Since the model output of choice is selected by populating the top gradient with a one-hot vector, there is no particular relationship to $f(\mathbf{x})$. Grad-CAM skips the SoftMax layer.

(Selvaraju et al., 2017) also propose *Guided Grad-Cam* and multiply the (upsampled) class-discriminative activation maps gained from Grad-CAM with the more finely-grained response maps from Guided Backpropagation element-wise, to obtain class discriminative heatmaps at a higher resolution.

Excitation Backpropagation (J. Zhang et al., 2018) (EB) models the attention of CNNs with ReLU activation units as top-down probabilistic Winner-Takes-All (Tsotsos et al., 1995) process with the aim to identify task-relevant neurons of a network. The method is quite similar to LRP, as it combines bottom-up information (inputs and weights) with backpropagated quantities (excitation maps, simlar to relevance) into the computation of attention maps. In fact, EB shares important assumptions with Deep Taylor Decomposition – namely the positivity of all layer inputs – and makes use of a backpropagation rule which is identical to the $\alpha\beta$-rule of LRP with $\alpha = 1, \beta = 0$. Attention maps can be used to localize input features responsible for the activation of (certain) output neurons and is closely related to CAM. EB skips the SoftMax layer.

DeepLift (Shrikumar et al., 2016; Shrikumar et al., 2017) computes contribution scores for the neurons of a network, by comparing its activations to a *reference* activation. Computing

the differences to reference activations effectively disables bias neurons (since bias activations are constant) and bias-like jumps in activation functions otherwise causing discontinuities in gradient-based explanation methods. Similar to (naive) Taylor Decomposition, DeepLift requires one (or several) explicitly given reference input(s) for computing explanation maps. Apart from computing contributions wrt to reference activations, DeepLift (optionally) follows the philosophy behind the $\alpha\beta$-rule of LRP, by considering the separate treatment of positive and negative layer inputs and forward activations. Furthermore, DeepLift – like LRP – resolves non-linearities by computing an output-input ratio for activation functions, thus avoiding the zero-gradient problem for saturated activation units. It can be shown that the method computes backward passes identical to LRP, given a network without or with only ReLU non-linearities, no bias neurons and the choice of a reference point causing all hidden neuron reference activations to be zero (Ancona et al., 2018). DeepLift skips the SoftMax layer.

PatternAttribution (Kindermans et al., 2017b) (PA) builds upon PN and the interaction idea behind LRP, by exposing the estimated signal computed via PA to the weight parameters of the model. It can be seen as an extension to DTD that learns from data how to choose root points $\widetilde{\mathbf{x}}$ for taylor expansion optimally. The explanations computed by PA are then for each neuron of the model (and the intput) its contribution to the predictor output $f(\mathbf{x})$ wrt to the estimated signal $\mathbf{s}$, absent the distractor $\mathbf{d}$.

Sampling- and Learning-based Interpretation

Sampling based interpretability approaches measure the importance of features of a given input $\mathbf{x}$ by provoking changes in $f(\mathbf{x})$ or approximating $f(\mathbf{x})$ via the exploration of the vicinity around $\mathbf{x}$. An advantage of these methods is the independance from type and architecture of the analyzed model, which comes at the price of computational inefficiency: Outside of this category, all other discussed analysis methods (except LRP and SA) are limited to DNN type models fulfilling very specific assumptions (*e.g.* ReLUs, max-pooling, GAP,...). Instead of computing an explanation as a backward pass through the model, these approaches rely on frequent (re)sampling of reference points $\widetilde{\mathbf{x}}$ and (re)evaluation of f, making them infeasible choices for large scale evaluations of classifier behaviour, such as presented in Chapter 6.

Probably the most simple, *occlusion*-based variant of sampling-based prediction analysis relies on local distortions of the input image in order to measure the importance of parts of the input locally (Zeiler et al., 2014) as Δf. Here, the size, frequency of application and type of occlusion masks used may have an effecton the resulting heatmaps (Ancona et al., 2018).

Local Interpretable Model-agnostic Explanations (Ribeiro et al., 2016) (LIME) uses sampling in a vicinity around a sample of interest $\mathbf{x}$ in order to compute local estimations of the predictor f from a family of *interpretable* functions $G = \{g\}$, such as linear models or decision trees. That is, LIME first derives *interpretable features* $\mathbf{x}'$ encoding human-understandable features from the data as *i.e.* binary vectors from the original inputs $\mathbf{x}$, regardless of the actual complexity of $\mathbf{x}$. Then an interpretable function g is fit onto $\mathbf{x}'$ such that it emulates the behaviour of f on $\mathbf{x}$ based on a sampling procedure in a neighborhood of $\mathbf{x}$. The paper presents results for sparse linear models implementing the function g and reports run times under 3 seconds for explaining random forests with 1000 trees and 5000 random samples used for fitting the explainable model. For inception-type DNNs for image classification such as the GoogleNet, the explanation of single predictions takes ≈ 10 minutes.

Prediction Difference Analysis (Zintgraf et al., 2016; Zintgraf et al., 2017) (PDA) visualizes the response of a DNN (or any single neuron) to a specific input and thus identifies areas of the (image) input speaking for or against a target class. PDA is similar to the occlusion-based approach from (Zeiler et al., 2014) but instead of occluding parts of the input, PDA *removes* information by marginalizing each feature out to measure its importance to $f(\mathbf{x})$. Since rigorous marginalization can for complex models become computationally infeasible, the authors use conditional sampling within the pixel neighborhood of an analyzed feature as an approximate alternative to keep analysis times within feasible bounds. The authors report analysis times for one image from the ImageNet dataset to be on average 20 mininutes for AlexNet, 30 minuntes for the GoogLeNet and 70 minutes for the VGG-16 model.

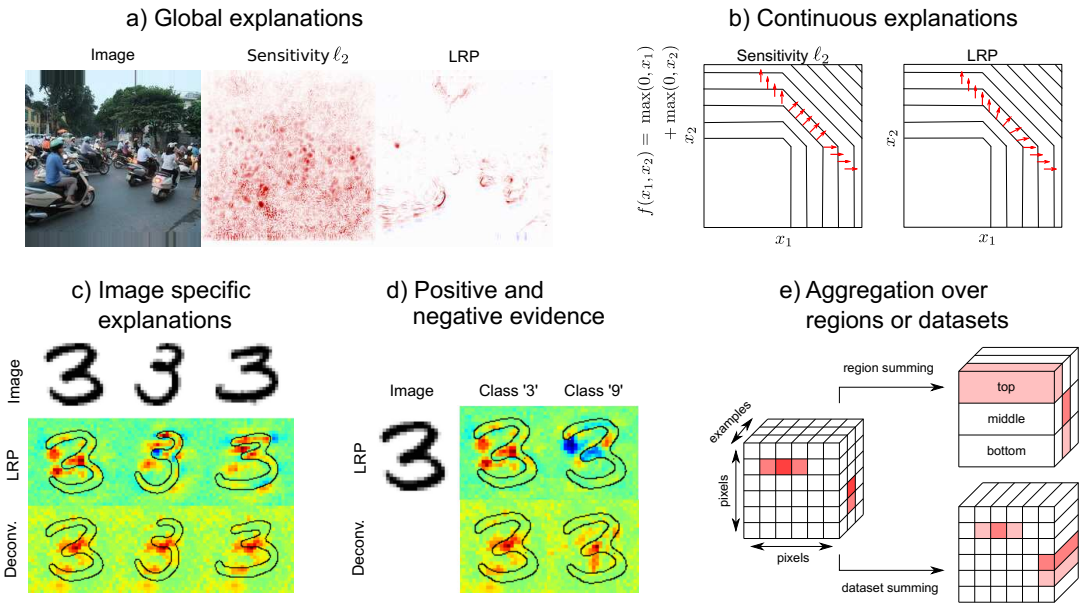

FIGURE 3.3: **Desirable properties of heatmapping methods.** *a)* LRP provides global explanations, *i.e.* , indicates what are the features that explain the prediction "scooter", whereas Sensitivity Analysis provides local explanations, *i.e.* , shows what would make the image less or more belong to the category "scooter". *b)* LRP provides continuous explanations (red arrows) which change slowly with the input, whereas the explanations provided by Sensitivity Analysis change abruptly due to discontinuities. *c)* LRP provides image specific explanations, because it takes into account the weights *and* the activations, whereas the Deconvolution method only considers the weights and thus produces the same heatmaps for different input images in the absence of max-pooling layers. *d)* LRP distinguishes between positive evidence supporting a prediction (red) and negative evidence speaking against it (blue), whereas Deconvolution does not provide signed explanations. *e)* The conservation property of LRP provided a meaningful normalization of the heatmaps, thus allows one to aggregate the explanations over regions or datasets.

(Fong et al., 2017) proposes a general framework for learning different kinds of explanations for any type of black box classifier. The paper argues that the meaning of an explanation depends on the right kind of variation applied to an input of interest. The method thus can consider the interpretation of a prediction wrt to chosen types of perturbation (*e.g.* sampling strategies). The optimization of an explanation is based on a stochastic gradient descent type algorithm for finding the (smallest) maximally informative image regions by deletion and preservation of information. In contrast to other methods, the proposed framework directly edits the input image of the explained prediction in an informative way. The approach proposed in (Fong et al., 2017) is roughly one order of magnitude less computationally expensive (in terms of required iteration steps) compared to LIME.

3.3 Theoretical Analysis of Desirable Heatmap Properties

In the following we investigate the advantages and limitations of the heatmapping methods presented in detail. We show that LRP has six desirable properties, which are not or only partly satisfied by Sensitivity Analysis and the Deconvolution method.

3.3.1 Global Explanations

A desired property of heatmapping methods is that they provide global explanations, *e.g.* indicate what are the features that compose a given car. This property is satisfied by LRP

and to some extent by the Deconvolution method, but not by Sensitivity Analysis. The latter gives for every pixel a direction in RGB-space in which the prediction increases or decreases, but it does not indicate directly whether a particular region contains evidence for or against the prediction made by a classifier. Thus, it provides *local* explanations, *e.g.* indicate where change would make a given car look more/less like a car. The subplot *a)* of Figure 3.3 shows the qualitative difference between gradient- and LRP-type explanations. In terms of gradients it is a valid explanation to put high norms on the empty street, because there exists a direction in the input space in which the classifier prediction can be increased by putting motor-bike like structures in there. However, from a global explanation perspective the streets are not very indicative of the class scooter in this particular image. This example shows that regions consisting of pure background may have a notable sensitivity, which makes gradient-type explanations noisier than LRP and Deconvolution heatmaps (also see Figure 3.6).

3.3.2 Continuous Explanations

Another desired property of heatmapping methods is that they provide continuous explanations, *i.e.* , small variations in the input should not result in large changes in the heatmap. Also this property is not satisfied by gradient-type methods. The multiplication of the signal by an indicator function in the rectification layer (see Section 3.2.1) makes the backward mapping of gradient-type methods discontinuous, *i.e.* , they may abruptly switch from considering one feature as being highly relevant to considering another feature as being the most important one. The subplot *b)* of Figure 3.3 shows that Sensitivity Analysis of a two-dimensional function $f(x_1, x_2)$ results in discontinuous explanations (red arrows abruptly change direction), whereas LRP (and also Deconvolution) does not show this behavior and thus provides more reliable explanations.

3.3.3 Image Specific Explanations

Salient features represent average explanations of what distinguishes one image category from another. For individual images these explanations may be meaningless or even wrong. The Deconvolution method only implicitly takes into account properties of individual images through the unpooling operation. The backprojection over filtering layers is independent of the individual image. Thus, when applied to Neural Networks without a pooling layer, both methods will not provide individual (image specific) explanations, but rather average salient features. LRP's rule for filtering layers on the other hand takes into account *both* filter weights and lower-layer neuron activations. This allows for individual explanations even in a Neural Network without pooling layers. The subplot *c)* of Figure 3.3 demonstrates this property on a simple example. We compare the explanations provided by the Deconvolution method and LRP for a Neural Network without pooling layers trained on the MNIST data set. One can see that LRP provides individual explanations for all images in the sense that when the digit in the image is slightly rotated, then the heatmap adapts to this rotation and highlights the relevant regions of this particular rotated digit. The Deconvolution heatmap on the other hand is not image-specific because it only depends on the weights and not the neuron activations. If pooling layers were present in the network, then the Deconvolution approach would implicitly adapt to the specific image through the unpooling operation. Still we consider this information important to be included when backprojecting over filtering layers, because neurons with large activations for a specific image should be regarded as more relevant, thus should backproject a larger share of the relevance.

3.3.4 Positive and Negative Evidence

In contrast to Sensitivity Analysis and the Deconvolution Method, LRP provides signed explanations and thus distinguishes between positive evidence, supporting the classification decision, and negative evidence, speaking against the prediction. The subplot *d)* of Figure 3.3 shows that LRP responses can be well interpreted in that way; the red and blue color represent positive and negative evidence, respectively. In particular, when backpropagating the (artificial) classification decision that the image has been classified as "9", LRP provides a

very intuitive explanation, namely that in the left upper part of the image the missing stroke closing the loop (blue color) speaks against the fact that this is a "9" whereas the missing stroke in the left lower part of the image (red color) supports this decision. The Deconvolution method (and the gradient-type explanation) does not allow such interpretation.

3.3.5 Aggregating over Regions or Datasets

Aggregating explanations over image regions or over different datasets (see subplot *e)* of Figure 3.3) is a desired property requiring meaningful normalization of the pixel-wise scores. The explanations computed by Sensitivity Analysis and the Deconvolution method are not normalized (naturally), so that aggregation may lead to meaningless results (*e.g.* when the heatmap of one image has values which are an order of magnitude larger than values of other heatmaps). The LRP scores on the other hand are directly related to the classification output through the conservation principle, thus are meaningfully normalized. This allows for a meaningful aggregation.

3.3.6 Relation to Classification Output

Another desirable property of heatmapping methods is an explicit mathematical relation between heatmap and the classification output, because this allows for an interpretation of the obtained scores. Such explicit relation exists for the gradient-type approach and for the LRP method (see formula in Figure 3.3). For the Deconvolution method these relationships cannot be expressed analytically, because negative evidence ($R^{(l+1)} < 0$) is discarded during the backpropagation due to the application of the ReLU function and the backward signal is not normalized layer-wise, so that few dominant $R^{(l)}$ may largely determine the final heatmap scores.

Finally, an additional justification for the way LRP technically operates can be found in (Montavon et al., 2017) where the method is shown for certain choices of parameters to perform a *Deep Taylor Decomposition* of the Neural Network function.

3.4 Evaluating Heatmaps

In this section, we introduce a set of methods to evaluate empirically the quality of a heatmapping technique. Although humans are able to intuitively assess the quality of a heatmap by matching with prior knowledge and experience of what is regarded as being relevant, defining objective criteria for heatmap quality is very difficult. In this chapter we refrain from mimicking the complex human heatmap evaluation process which includes attention, interest point models and perception models of saliency (Heeger et al., 1996; Itti et al., 2000; Simoncelli et al., 2001; Itti et al., 2001) for the reason that we are interested in finding those regions that are relevant for a given classifier, not a human assessor.

Rather than representing human reasoning, or matching some ground truth on what is important in an image, heatmaps should reflect the machine learning classifier's own "view" on the classification problem, more precisely, identify the pixels used by the classifier to support its decision. Thus, heatmap quality does not only depend on the algorithms used to compute a heatmap, but also on the performance of the classifier, whose efficiency largely depends on the model being used, and the amount and quality of available training data. For example, if the training data does not contain images of the digits "3", then the classifier can not know that the absence of strokes in the left part of the image (see example in Figure 3.1) is important for distinguishing the digit "3" from digits "8" and "9". Thus, explanations can only be as good as the data provided to the classifier. A random classifier will lead to uninformative heatmaps.

Note also that a heatmap differs from a segmentation mask (see Figure 3.1) in several ways: First, a heatmap can rightfully associate relevance to pixels outside the object to detect, for example, when the context of the object (*e.g.* water texture behind a boat) provides some useful information for classification. Second, segmentation masks are binary (*i.e.* they essentially determine whether a pixel is part or not of the object to detect). A heatmap, on

the other hand, provides a gradation of pixel scores, which correspond to the degree of importance of each pixel for determining predicted class membership. Points of interest (*e.g.* eyes, or small objects), might concentrate as much information for determining the class as larger surfaces. A heatmap can also associate negative values to pixels that contradict the prediction.

3.4.1 Heatmap Evaluation via Input Perturbations

More formally, a heatmap is an array of pixel-wise scores $(R_p)_p$, that indicates which pixels are *important* for a classification decision (*e.g.* which pixels make $f(\mathbf{x})$ large). A heatmap can be viewed as defining a subspace composed of pixels with high importance scores, on which the function $f(\mathbf{x})$ must be scrutinized. We expect the pixels to which most relevance is associated to be those that are the most likely to affect the value $f(\mathbf{x})$ if they are perturbed, in other words, we would like to test how fast the function value drops when moving in this subspace.

To test this expected behavior, we consider a greedy iterative procedure that consists of measuring how the class encoded in the image (*e.g.* as measured by the function f) *disappears* when we progressively *remove* information from the image $\mathbf{x}$, a process referred to as *input perturbation* or *region perturbation*, at the specified locations. The method is a generalization of the approach presented in (S. Bach et al., 2015), where the perturbation process is a state flip of the associated binary pixel values (single pixel perturbation). The method that we propose here applies more generally to *any* set of locations (*e.g.* local windows) and *any* local perturbation process such as local randomization or blurring.

We define a heatmap as an ordered set of locations in the image, where these locations might lie on a predefined grid.

$$\mathcal{O} = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_L) \tag{3.4}$$

Each location $\mathbf{r}_p$ is for example a two-dimensional vector encoding the horizontal and vertical position on a grid of pixels. The ordering can either be chosen at hand, or be induced by a heatmapping function $h_p = \mathcal{H}(\mathbf{x}, f, \mathbf{r}_p)$, typically derived from a class discriminant f (see methods in Section 3.2). The scores $\{h_p\}$ indicate how important the given location $\mathbf{r}_p$ of the image is for representing the image class. The ordering induced by the heatmapping function is such that for all indices of the ordered sequence $\mathcal{O}$, the following property holds:

$$(i < j) \Leftrightarrow \left(\mathcal{H}(\mathbf{x}, f, \mathbf{r}_i) > \mathcal{H}(\mathbf{x}, f, \mathbf{r}_j)\right) \tag{3.5}$$

Thus, locations in the image that are most relevant for the class encoded by the classifier function f will be found at the beginning of the sequence $\mathcal{O}$. Conversely, regions of the image that are mostly irrelevant will be positioned at the end of the sequence.

We consider a region perturbation process that follows the ordered sequence of locations. We call this process *most relevant first*, abbreviated as MoRF. The recursive formula is:

$$\begin{aligned} \mathbf{x}_{\text{MoRF}}^{(0)} &= \mathbf{x} \\ \forall\, 1 \leqslant k \leqslant L:\ \mathbf{x}_{\text{MoRF}}^{(k)} &= g\left(\mathbf{x}_{\text{MoRF}}^{(k-1)}, \mathbf{r}_k\right) \end{aligned} \tag{3.6}$$

where the function g removes information of the image $\mathbf{x}_{\text{MoRF}}^{(k-1)}$ at a specified location $\mathbf{r}_k$ (*i.e.* a single pixel or a local neighborhood) in the image. Figure 3.4 shows the effect of above perturbation approach to input digits from the MNIST (LeCun, 1998) dataset following the experimental setup used in (S. Bach et al., 2015).

Throughout the paper we use a function g which replaces all pixels in a $(m \times m)$ neighborhood around $\mathbf{r}_k$ by values sampled randomly from a uniform distribution. The choice of uniform distribution follows our intention to use a model-free method to generate perturbations. A model-based estimation of probabilities for patches may result in biases due to the model assumptions. Using the uniform distribution ensures that we treat all regions equally and explore the whole imaging space. Furthermore it ensures that we evaluate the behaviour of the classifier under perturbations that are off the data-manifold. We consider a region highly relevant if replacing the information in this region in *arbitrary* ways reduces the

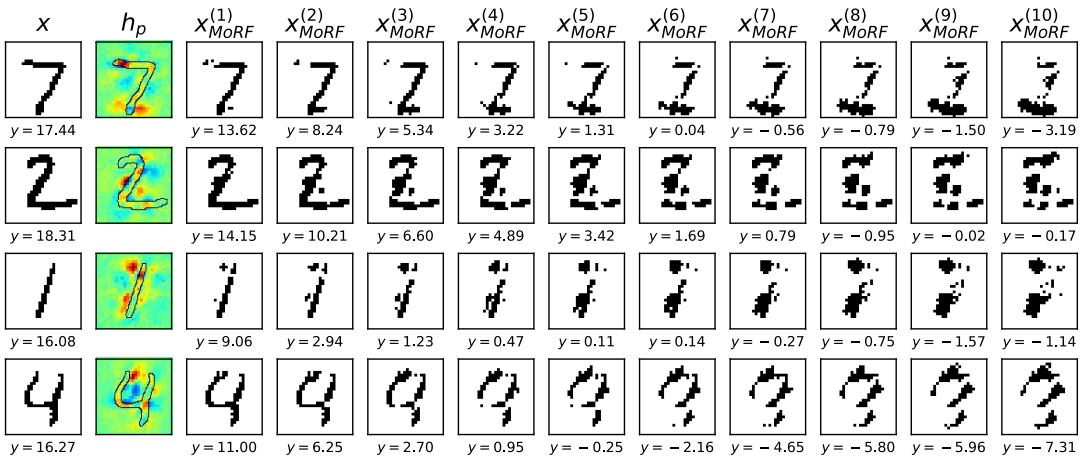

FIGURE 3.4: **Example input perturbations on MNIST digits.** The two leftmost columns show the original input digit with corresponding relevance maps for the tanh-activated model described in Section 2.3.3. The remaining columns shown the perturbed digits after up to 10 perturbation steps. The value y below each digit shows the logit output of the class predicted for the unperturbed input $\mathbf{x}$. The model accepts inputs normalized to a value range of $[-1, 1]$ per pixel. For this example, the input images are thresholded at 0, assigning the value 1 to all foreground pixels (black) and -1 to all background pixels (white). Each perturbation step inverts the sign of 1% of the input pixels, following the algorithm described in Equation (3.6).

prediction score of the classifier; we do not want to restrict the analysis to highly specialized information removal schemes.

When comparing different heatmaps using a fixed $g(\mathbf{x}, \mathbf{r}_k)$ our focus is typically only on the highly important regions (*i.e.* the sorting of the h_p values on the non-relevant regions is not important). The quantity of interest in this case is the area over the MoRF perturbation curve (AOPC):

$$\text{AOPC} = \frac{1}{L+1}\left\langle \sum_{k=0}^{L} f\left(\mathbf{x}^{(0)}_{\text{MoRF}}\right) - f\left(\mathbf{x}^{(k)}_{\text{MoRF}}\right) \right\rangle_{p(\mathbf{x})} \tag{3.7}$$

where $\langle\cdot\rangle_{p(\mathbf{x})}$ denotes the average over all images in the data set. An ordering of regions such that the most sensitive regions are ranked first implies a steep decrease of the graph of MoRF, and thus a larger AOPC.

3.5 Experimental Results

In this section we use the proposed heatmap evaluation procedure to compare heatmaps computed with the LRP algorithm, the Deconvolution approach (Zeiler et al., 2014) (Section 3.2.2) and the sensitivity-based method (Simonyan et al., 2013) (Section 3.2.1) to a random order baseline. Exemplary heatmaps produced with these algorithms are displayed and discussed in Section 3.5.3. At the end of this section we briefly investigate the correlation between heatmap quality and network performance.

3.5.1 Experimental Setup

We demonstrate the results on a classifier for the MIT Places data set (Zhou et al., 2014b) provided by the authors of this data set and the Caffe reference model (Jia et al., 2014) for ImageNet. We kept the classifiers unchanged. They are both near state of the art convolutional neural networks and consist of layers of convolution, ReLU and max-pooling neurons. Both classifiers share the same architecture proposed in (Krizhevsky et al., 2012), namely a

sequence of

$$\begin{aligned}
&\texttt{Conv} \rightarrow \texttt{ReLU} \rightarrow \texttt{Local Norm} \rightarrow \texttt{Max-Pool} \rightarrow \\
&\texttt{Conv} \rightarrow \texttt{ReLU} \rightarrow \texttt{Local Norm} \rightarrow \texttt{Max-Pool} \rightarrow \\
&\texttt{Conv} \rightarrow \texttt{ReLU} \rightarrow \texttt{Conv} \rightarrow \texttt{ReLU} \rightarrow \texttt{Conv} \rightarrow \texttt{ReLU} \rightarrow \\
&\texttt{Max-Pool} \rightarrow \texttt{FC} \rightarrow \texttt{ReLU} \rightarrow \texttt{FC} \rightarrow \texttt{ReLU} \rightarrow \texttt{FC}
\end{aligned}$$

The Places classifier was trained on 2,448,873 randomly select images from 205 categories and the Caffe reference model was trained on 1.2 million images of ImageNet. The MIT Places classifier is used for two testing data sets. Firstly, we compute the AOPC values over 5040 images from the MIT Places testing set. Secondly, we use AOPC averages over 5040 images from the SUN397 data set (Xiao et al., 2010) as it was done in (Zhou et al., 2014a). We ensured that the category labels of the images used were included in the MIT Places label set. Furthermore, for the ImageNet classifier we report results on the first 5040 images of the ILSVRC2012 data set. The heatmaps are computed for all methods for the predicted label, so that our perturbation analysis is a fully unsupervised method during test stage.

Perturbation is applied to (9×9) non-overlapping regions each covering 0.157% of the image. This region size was selected, because (1) over 100 perturbation steps[3] it allows to perturb a significant part of the image (15.7%) and (2) it approximately matches the size of convolution filters used by the trained neural network models. For completeness, Appendix B.1 contains results for additional region sizes in Figure B.1. We replace all pixels in a region by randomly sampled (from uniform distribution) values.

In order to reduce the effect of randomness we repeat the process 10 times. For each ordering we perturb the first 100 regions, resulting for the (9×9) neighborhood in 15.7% of the image being exchanged. Running the experiments for 2 configurations of perturbations, each with 5040 images, takes roughly 36 hours on a workstation with 20 (10×2) Xeon HT-Cores. Given the above running time and the large number of configurations reported here, we considered the choice of 5040 images as sample size a good compromise between the representativity of our result and computing time.

3.5.2 Quantitative Comparison of Heatmapping Methods

We quantitatively compare the quality of heatmaps generated by the three algorithms described in Section 3.2. As a baseline we also compute the AOPC curves for random heatmaps (*i.e.* random ordering 0). Figure 3.5 displays the AOPC values as function of the perturbation steps (*i.e.* L from Equation (3.6)) relative to the random baseline.

From the figure one can see that heatmaps computed by LRP have the largest AOPC values, *i.e.* they better identify the relevant (wrt the classification tasks) pixels in the image than heatmaps produced with Sensitivity Analysis or the Deconvolution approach. This holds for all three data sets. The ϵ-LRP formula (see Equation (2.22)) performs slightly better than $\alpha\beta$-LRP (see Equation 2.26), however, we expect both LRP variants to have similar performance when optimizing for the parameters (here we use the same settings as in (S. Bach et al., 2015)).

The Deconvolution method performs as closest competitor and significantly outperforms the random baseline. Since LRP distinguishes between positive and negative evidence and normalizes the scores properly, it provides less noisy heatmaps than the Deconvolution approach (see Section 3.5.3) which results in better quantitative performance. As stated above Sensitivity Analysis targets a slightly different problem and thus provides quantitatively and qualitatively suboptimal explanations of the classifier's *decision*. Sensitivity provides local explanations, but may fail to capture the global features of a particular class. In this context see also the works of (Szegedy et al., 2013; Goodfellow et al., 2014; Nguyen et al., 2015) in which changing an image as a whole by a minor perturbation leads to a flip in the class labels, and in which rainbow-colored noise images are constructed with high classification accuracy.

Perturbation changes on the ILSVRC2012 dataset have a higher quantitative impact to the $f(\mathbf{x})$ of the model, according to our AOPC measure, than on the other two datasets. One

[3] For computational reasons we restricted the number of perturbation steps to 100.

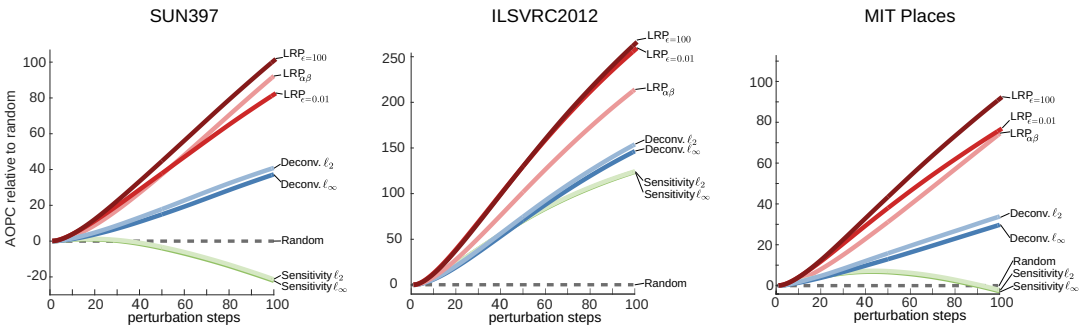

FIGURE 3.5: **Comparison of the Sensitivity Analysis, Deconvolution and LRP methods relative to the random baseline.** The LRP algorithms have largest AOPC values, *i.e.* best explain the classifier's decision, for all three data sets.

reason for this is that the ILSVRC2012 images contain more objects and less cluttered scenes than images from the SUN397 and MIT Places data sets, *i.e.* it is easier (also for humans) to capture the relevant parts of the image, which are also by nature located on comparatively smaller regions within the images. Also the AOPC difference between the random baseline and the other heatmapping methods is much smaller for the latter two data sets than for ILSVRC2012, because cluttered scenes contain evidence almost everywhere in the image whereas the background is less important for object categories.

An interesting phenomenon is the performance difference of Sensitivity heatmaps computed on SUN397 and MIT Places data sets, in the former case the AOPC curve of Sensitivity heatmaps is even below the curve computed with random ranking of regions, whereas for the latter data set the Sensitivity heatmaps are (at least initially) clearly better. Note that in both cases the same classifier (Zhou et al., 2014b), trained on the MIT Places data, was used. The difference between these data sets is that SUN397 images lie outside the data manifold (*i.e.* images of MIT Places used to train the classifier), so that partial derivatives need to explain local variations of the classification function $f(\mathbf{x})$ in an area in image space where f has not been trained properly. This effect is less strong for the MIT Places test data, as they are closer to the images used to train the classifier. Since both LRP and Deconvolution provide global explanations, they are less affected by this off-manifold testing.

We performed above evaluation also for both Caffe networks in training phase, in which the dropout layers were active. The results are qualitatively the same to the ones shown above. The LRP algorithm, which was explicitly designed to explain the classifier's decision, performs significantly better than the other heatmapping approaches. We would like to stress that LRP does not artificially benefit from the way we evaluate heatmaps as region perturbation is based on a assumption ("good" heatmaps should rank pixels according to importance wrt to classification) which is independent of the relevance conservation principle that is used in LRP. Note that LRP was originally designed for binary classifiers in which $f(\mathbf{x}) = 0$ denotes maximal uncertainty about prediction. The classifiers used here were trained with a different multiclass objective, namely that it suffices for the correct class to have the highest score. One can expect that in such a setup the state of maximal uncertainty is given by a positive value rather than $f(\mathbf{x}) = 0$. In that sense the setup here slightly disfavours LRP. However we refrained from retraining because it was important for us, firstly, to use classifiers provided by other researchers in an unmodified manner, and, secondly, to evaluate the robustness of LRP when applied in the popular multi-class setup.

3.5.3 Qualitative Comparison of Heatmapping Methods

In Figure 3.6 the heatmaps of four images of each data set are visualized. Red color indicates large scores, blue color indicates negative scores (only for LRP). The quantitative result presented above are in line with the subjective impressions. The Sensitivity and Deconvolution heatmaps are noisier and less sparse than the heatmaps computed with the LRP algorithm, reflecting the results obtained in Section 3.5.2. For SUN 397 and MIT Places the Sensitivity

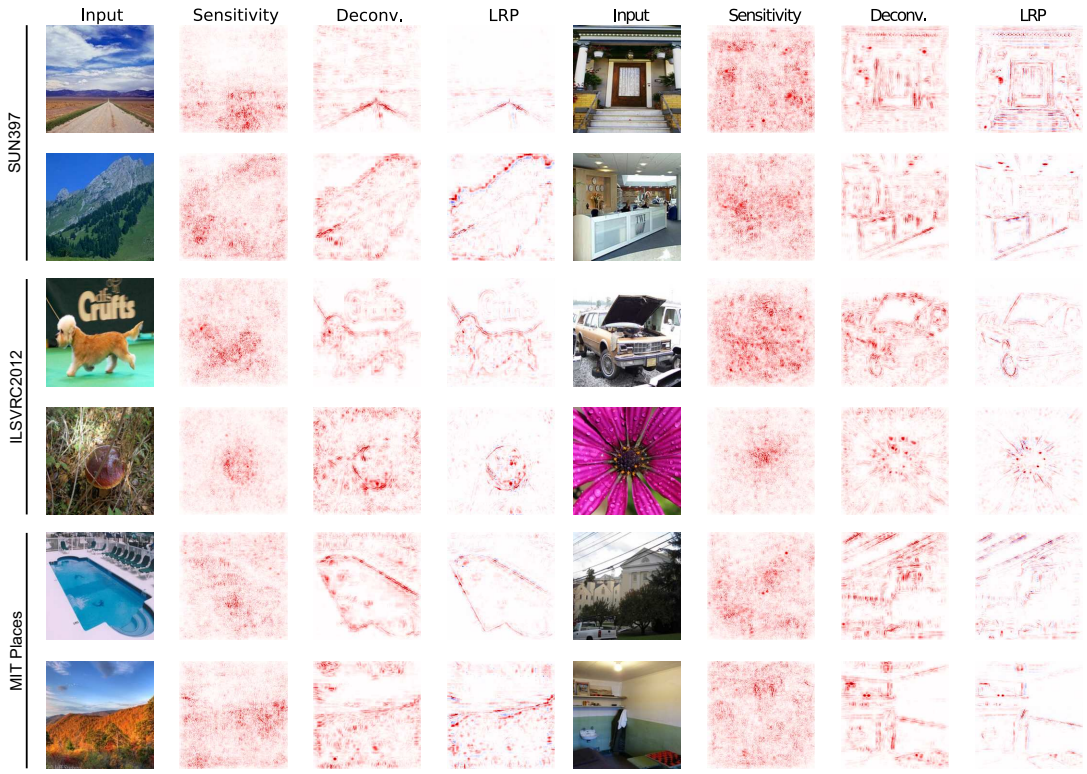

FIGURE 3.6: **Qualitative comparison of the three heatmapping methods for four exemplary images of the SUN397, ILSVRC2012 and MIT Places dataset.** Red color indicates large scores, blue color indicates negative scores (LRP only). The heatmaps computed with the LRP algorithm focus on the relevant features of the object class (*e.g.* face of the dog or volcano shape), whereas the Sensitivity and Deconvolution heatmaps are nosier and less focused. These qualitative observations are in line with the quantitative results in Figure 3.5. Figures B.2, B.3 and B.4 in the Appendix B.2 show additional results for qualitative comparison.

Heatmaps are close to random, whereas both LRP and Deconvolution highlight some structural elements in the scene (*e.g.* mountain shape for the class "volcano" or the arch shape for the class "abbey").

We remark that this bad performance of Sensitivity heatmaps does not contradict results like (Szegedy et al., 2013; Goodfellow et al., 2014). In the former works, an image gets modified as a whole, while in this work we are considering the quality of selecting local regions and ordering them. Furthermore gradients require to move in a very particular direction for reducing the prediction while we are looking for most relevant regions in the sense that changing them in any kind (*i.e.* randomly) will likely destroy the prediction. The Deconvolution and LRP algorithms capture more global (and more relevant) features than the sensitivity approach. Figures B.2, B.3 and B.4 in the Appendix B.2 show additional results for qualitative comparison.

3.5.4 Heatmap Quality and Neural Network Performance

In the last experiment we briefly show that the quality of a heatmap, as measured by AOPC, provides information about the overall DNN performance. The intuitive explanation for this is that well-trained DNNs much better capture the relevant structures in an image, thus produce more meaningful heatmaps than poorly trained networks which rather rely on global image statistics. Thus, by evaluating the quality of a heatmap using the proposed procedure we can potentially assess the network performance, at least for classifiers that were based on the same network topology. Note that this procedure is based on perturbation of the input

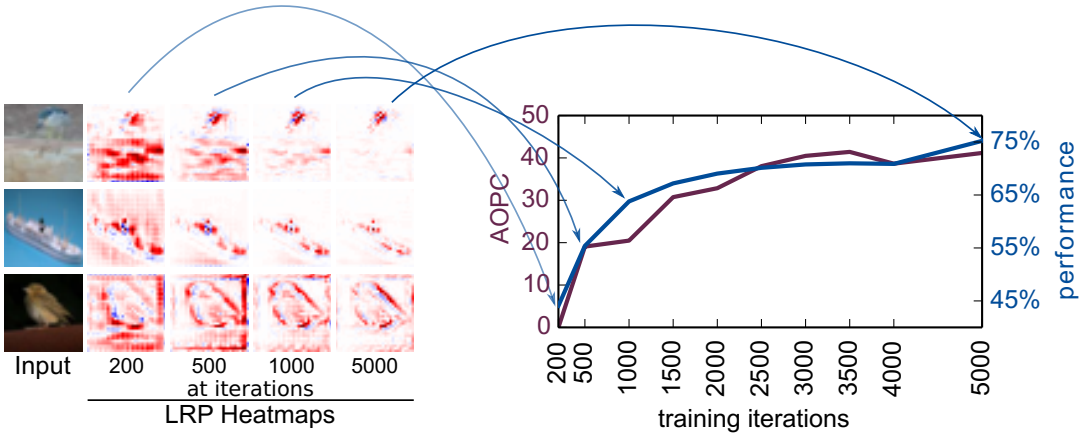

FIGURE 3.7: **Evaluation of network performance by using AOPC values obtained with LRP on CIFAR-10.**

of the classifier with the highest predicted score. Thus this evaluation method is purely unsupervised and does not require labels of the testing images. Figure 3.7 depicts the AOPC values obtained with LRP and the performance for different training iterations of a DNN for the CIFAR-10 data set (Krizhevsky, 2009). We did not perform these experiments on a larger data set since the effect can still be observed nicely in this modest data size. The correlation between both curves indicates that heatmaps contain information which can potentially be used to judge the quality of the network. This chapter did not indent to profoundly investigate the relation between network performance and heatmap quality, this is a topic for future research.

3.6 Limitations

This chapter demonstrates that each of the closely inspected analysis methods – Sensivity Analysis, the Deconvolution Method and LRP – reveals different aspects and properties of a prediction made by a Neural Network classifier. While we have shown that the heatmaps obtained from LRP best represent the *decision* of the classifier in terms of its interaction with the input, this means that the use of relevance maps alone might not be enough to fully understands *all aspects* of a model decisions. So does LRP for example explain how the model uses information from an input $\mathbf{x}$ to arrive at its decision, while Sensitivity Analysis informs about the sensibility (or vulnerability) of the model to changes at $\mathbf{x}$, and the Deconvolution method reveals which input patterns provoke the activation of the analyzed output (for suitable network architectures). A combination of orthogonal methods is required to obtain a *comprehensive* understanding of a prediction for a given input point $\mathbf{x}$.

Appendix B.1 provides additional and more extensive results for the perturbation experiments discussed in this chapter. While the evaluations wrt to different perturbation block sizes consolidate the quality of heatmaps derived with LRP, it is clear that choices for shape and size of the perturbed areas in input space, as well as the way the inputs are perturbed, have an effect on the measured results. Since the model reacts to these free parameters of our perturbation framework, more extensive evaluations over a range of parameters should be considered for future assessments. This, of course, comes at an increased computational effort. For the special case of the Fisher vector model discussed in Chapter 4 we can come up with a more elegant strategy for perturbing the input samples (see Appendix C.2), which is invariant to choices for perturbation mask size and replacement strategy, but also highly situational (wrt to the design of the model) and thus does applicable as a general solution for other predictors.

3.7 Conclusion

Research on DNNs has been traditionally focusing on improving the quality, algorithmics or the speed of a Neural Network model. We have so far studied an orthogonal research direction, namely, we have contributed to furthering the understanding and transparency of the decision making implemented by a trained DNN: For this we have focused on the heatmap concept that, *e.g.* in a computer vision application, is able to attribute the contribution of individual pixels to the DNN inference result for a novel data sample. While heatmaps allow a better intuition about what has been learned by the network, we tackled the so far open problem of quantifying the quality of a heatmap. In this manner different heatmap algorithms can be compared quantitatively and their properties and limits can be related. We proposed a region perturbation strategy that is based on the idea that flipping the most salient pixels first should lead to high performance decay. A large AOPC value as a function of perturbation steps was shown to provide a good measure for a very informative heatmap. We also showed quantitatively and qualitatively that Sensitivity maps and heatmaps computed with the Deconvolution algorithm are much noisier than heatmaps computed with the LRP method, thus are less suitable for identifying the most important regions wrt the classification task. Above all we provided first evidence that heatmaps may be useful for assessment of Neural Network performance. Bringing this idea into practical application will be a topic of future research. Concluding, we have provided the basis for an accurate quantification of heatmap quality.

Note that a good heatmap can not only be used for better understanding of DNNs but also for a priorization of image regions. Thus, regions of an individual image with high heatmap values could be subjected to more detailed analysis. This could in the future allow highly time efficient processing of the data only *where it matters*.

Chapter 4

Comparing Fisher Vector SVMs and Deep Neural Networks

Fisher vector classifiers and Deep Neural Networks are popular and successful algorithms for solving image classification problems. However, both are generally considered "black box" predictors as the non-linear transformations involved have so far prevented transparent and interpretable reasoning. In the previous Chapter 3 we have shown quantitatively that relevance maps computed with LRP are representative of what parts of the input a learned model uses for prediction, compared to other methods. By design LRP follows a more general principle than other saliency methods and thus is applicable to a wider range of classifier types, beyond Neural Networks. These premises enable us to use LRP as a tool for analyzing the aforementioned Fisher vector predictor and Neural Network model contrastingly. We aim to gain insight into different prediction strategies employed by both predictor types on the same task, in order to find what makes the reasoning of Neural Networks (so much) better than the compared Fisher vector classifier's, a former state of the art model for image categorization.

In this chapter we extend the LRP framework also for Fisher vector classifiers and then use it as a tool for analysis to quantify the importance of image context for classification, qualitatively compare DNNs against FV classifiers in terms of important image regions and detect potential flaws and biases in data. All experiments are performed on the PASCAL VOC 2007 and ILSVRC 2012 data sets. This chapter covers the contributions from (Lapuschkin et al., 2016a) and adds in part novel analyses.

4.1 Introduction

While much of research is devoted to improve machine learning model performance or, *e.g.* , extending the applicability of DNNs to more domains (Hochreiter et al., 1997; Koutník et al., 2013; Karpathy et al., 2014a; Karpathy et al., 2014b; J. Zhang N. D. et al., 2014), we focus here on a different question, namely the impact of context, and the ability to use context. This question was raised already during times of the Pascal VOC challenge, where the amount of context was a matter of speculation[1].

The question of context is considered for two prominent types of classifiers. The first type, Fisher vectors (FV) (Perronnin et al., 2010; Sánchez et al., 2013) are based on computing a single feature map on an image as a whole and subsequently computing one score. In such a setup one can expect that context plays naturally a role for the prediction as the image is processed as a whole during training and testing. In case of small training sample sizes and the absence of opportunities for fine-tuning, Fisher vectors still might be a viable alternative to DNNs due to their reduced parameter space. Examples for performance issues of Deep Neural Networks on small sample sizes without finetuning can be seen in (Zeiler et al., 2014). The question of context is also open for the second type, Deep Neural Networks. One might assume that context plays no role for Neural Networks when they are used in classification by detection setups. For example, a recent ImageNet challenge winner relied on 144 crops per test image and classifier (Szegedy et al., 2015). Another work using Pascal VOC data (Oquab et al., 2014) used at test time 500 multi-scale patches per test image. However

[1] *c.f.* PASCAL VOC workshop presentation slides in (Everingham et al., 2010)

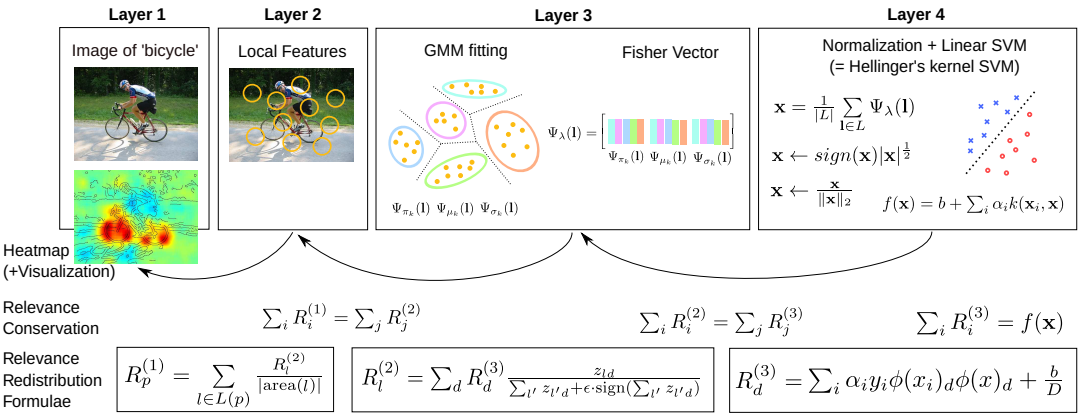

FIGURE 4.1: **Computing Fisher vector representation of an image and explaining the classification decision.**

in certain setups computing several hundred windows as required for classification by detection setups may not be possible, *e.g.* when using hardware without GPUs and much main memory, such as consumer laptops or smartphones, and when having time constraints for computation of the test prediction on an image. One can expect to see a larger impact of context when resorting to a few regions of an image at test time only, and thus training and testing with larger image patches.

Our contribution here is as follows.
(1) We extend Layer-wise Relevance Propagation and apply it for the first time to Fisher vectors.
(2) We define measures for the amount of context used for prediction in a single test image as a function of relevance and
(3) extend the perturbation-based approach for measuring heatmap qualities to Fisher vector representations. (4) We apply the measures of context for Neural Networks and Fisher vector based classifiers on the Pascal VOC dataset, as it offers a way to approximately validate context by its bounding box annotation. We compare the context dependence of Fisher vectors against Neural Networks which were trained on larger batches of input images.
(5) We show that this methodology is able to identify strong cases of context and biases in the training data even without using bounding box information and
(6) reveal a corellation between model complexity, prediction performance and the use of object-specific features instead of contextual image features.

The next section reviews related work. Section 4.2 briefly describes the Fisher vector classifier. Section 4.3 introduces the extended LRP method to decompose a Fisher vector prediction into scores for small regions of the order of a local feature. The same section also proposes a novel LRP-based measure of the importance of context. Section 4.5 introduces the experimental setup and presents results. The chapter concludes in Section 4.7 with a summary and an outlook.

4.2 Fisher Vectors Briefly Introduced

Fisher vectors (Perronnin et al., 2010; Sánchez et al., 2013) are a powerful tool to compute rich image or video representations and provide state of the art performance amongst feature extraction algorithms. Figure 4.1 summarizes the steps involved in computing FV representation of an image. We introduce here a notation which later will be used in the Section 4.3.

An integral part for computing FVs is to fit a Gaussian Mixture Model (GMM) on top of the local descriptors $L = \{l\}$ extracted from the training data to serve as a soft vocabulary of visual prototypes. Let us assume a K-component GMM $\lambda = \{(\pi_k, \mu_k, \Sigma_k)\}_{k=1..K}$ where π_k is the mixture weight of component k, with $\sum_k \pi_k = 1$ and $\forall k : \pi_k \geqslant 0$, μ_k is the mean vector of the kth mixture component and Σ_k its (diagonal) covariance matrix. For the

computation of a *complete*[2] FV representation of an image, each local descriptor l is related to all K components of the trained GMM in its 0th (soft mapping weight), 1st (deviation from mean) and 2nd moment (variance) (Sánchez et al., 2013), *i.e.*

$$\Psi_{\pi_k}(l) = \frac{1}{\sqrt{\pi_k}}\left(\gamma_k(l) - \pi_k\right) \tag{4.1}$$

$$\Psi_{\mu_k}(l) = \frac{1}{\sqrt{\pi_k}}\gamma_k(l)\left(\frac{l-\mu_k}{\sigma_k}\right) \tag{4.2}$$

$$\Psi_{\sigma_k}(l) = \frac{1}{\sqrt{\pi_k}}\gamma_k(l)\frac{1}{\sqrt{2}}\left(\frac{(l-\mu_k)^2}{\sigma_k^2} - 1\right) \tag{4.3}$$

with $\Psi_{\pi_k}(l) \in \mathbb{R}$, $\{l, \Psi_{\mu_k}(l), \Psi_{\sigma_k}(l)\} \in \mathbb{R}^D$ and $\gamma_k(l)$ returning the (scalar) soft assignment of l to the kth mixture component. The FV embedding $\Psi_\lambda(l)$ for a single descriptor l is then obtained by concatenating the mapping outputs for all K components into a $(1+2D)\cdot K$ dimensional vector

$$\Psi_\lambda(l) = \Big[\underbrace{\Psi_{\pi_1}(l)\,,\,\ldots\,,\Psi_{\pi_K}(l)}_{K\ \text{scalar values}},\ \underbrace{\Psi_{\mu_1}(l),\,\ldots\,,\Psi_{\mu_K}(l)}_{K\ \text{concatenated vectors in}\ \mathbb{R}^D}\ ,\ \underbrace{\Psi_{\sigma_1}(l),\,\ldots\,,\Psi_{\sigma_K}(l)}_{K\ \text{concatenated vectors in}\ \mathbb{R}^D}\Big] \tag{4.4}$$

Having computed all those (as we will refer to now as *unnormalized*) Fisher embeddings for all individual local descriptors, a single image-wise descriptor is obtained by averaging over the complete set of $\{\Psi_\lambda(l)|l \in L\}$, followed by power normalization to reduce the sparsity of the descriptor and ℓ_2-normalization which is known to prediction performance (Perronnin et al., 2010):

$$\begin{aligned}
\mathbf{x} &\leftarrow \frac{1}{|L|}\sum_{l\in L}\Psi_\lambda(l) && \text{mapping aggregation}\\
\mathbf{x} &\leftarrow \operatorname{sign}(\mathbf{x})|\mathbf{x}|^{\frac{1}{2}} && \text{power normalization}\\
\mathbf{x} &\leftarrow \frac{\mathbf{x}}{\|\mathbf{x}\|_2} && \ell_2\text{-normalization}
\end{aligned} \tag{4.5}$$

The application of both final normalization steps results in a so called *improved Fisher Kernel* and is – when combined with a linear SVM (Cortes et al., 1995) such as the FV model evaluated in this chapter – equivalent to the transformation of the unnormalized FV using the Hellinger's kernel function (Perronnin et al., 2010). *I.e.* expressing both above normalization steps in Equation (4.5) as a (kernel) feature mapping Φ taking as input the unnormalized FV $\mathbf{x}$ yields

$$\Phi(\mathbf{x}) = \frac{\operatorname{sign}(\mathbf{x})|\mathbf{x}|^{\frac{1}{2}}}{\|\operatorname{sign}(\mathbf{x})|\mathbf{x}|^{\frac{1}{2}}\|_2} = \frac{\operatorname{sign}(\mathbf{x})\sqrt{|\mathbf{x}|}}{\sqrt{\sum_d \underbrace{\operatorname{sign}(x_{(d)})^2}_{=1}|x_{(d)}|\underbrace{^{\frac{1}{2}\cdot 2}}_{=1}}} = \operatorname{sign}(\mathbf{x})\sqrt{\frac{|\mathbf{x}|}{\|\mathbf{x}\|_1}}$$

$$\implies k(\mathbf{x},\mathbf{y}) = \sum_d \Phi(\mathbf{x})_d\Phi(\mathbf{y})_d = \sum_d \operatorname{sign}(x_{(d)}y_{(d)})\underbrace{\sqrt{\frac{|x_{(d)}|}{\|\mathbf{x}\|_1}\cdot\frac{|y_{(d)}|}{\|\mathbf{y}\|_1}}}_{\text{Hellinger's kernel}}\,. \tag{4.6}$$

4.3 Deriving LRP for Fisher Vector Mappings

We extend the concept of Layer-wise Relevance Propagation for Bag of Words models as introduced in Chapter 2, Section 2.4 to the mappings defining the improved Fisher Kernel. Section 2.4 has so far assumed that BoW mappings are histograms and thus dominantly non-negative. For Fisher vectors this assumption does not hold, as the features are derivatives

[2] In contrast to a reduced set of output mappings, *e.g.* by omitting a component such as $\Psi_{\pi_k}(l)$ as done in (Chatfield et al., 2011) due to its low discriminative power (Sánchez et al., 2013).

with respect to parameters. We present an application of LRP to Fisher vector mappings, starting with the SVM classifier as a mapping of (kernelized) features

$$f(\mathbf{x}) = b + \sum_i \alpha_i y_i \sum_{d=1}^{D} \phi(\mathbf{x}_i)_d \phi(\mathbf{x})_d, \tag{4.7}$$

where $\mathbf{x}$ is a sum-aggregation of unnormalized Fisher vectors representing an image and $\Phi(\mathbf{x})$ is its normalization (Equation (4.6)). Since Φ is a component-wise operation, we can express the relevance scores $R_d^{(3)}$ attributed to each Fisher vector dimension d in consistency with Equations (2.72) and (2.74):

$$R_d^{(3)} = \sum_i \alpha_i y_i \phi(x_i)_d \phi(x)_d + \frac{b}{D}\,. \tag{4.8}$$

To further compute relevance scores $R_l^{(2)}$ for local descriptors l, it is required to express $\Psi_\lambda(l)$ in terms of mappings z_{ld} from local descriptors l to dimensions d in the Fisher vector feature space, as per Equation (2.78). Since the normalization steps Φ have already been resolved as part of the classification function, only the decomposition wrt to the mappings involved in the computation of the unnormalized Fisher vector remains. We can express the dth dimension of the unnormalized Fisher vector $\mathbf{x}$ as

$$x_d = \sum_{l \in L} m_d(l) = \sum_{l \in L} z_{ld}\,, \tag{4.9}$$

a sum over mappings of local features l into a scalar subspace of the feature space of Fisher vectors. Via the notation of Equation (4.4) in Section 4.2, z_{ld} is given as

$$\begin{aligned} z_{ld} &= \Psi_\lambda(l)_{(d)} \\ &= \begin{cases} \frac{(\gamma_k(l) - \pi_k)}{\sqrt{\pi_k}} & ;\ d = k,\ k \in [1, K] \\ \frac{\gamma_k(l)}{\sqrt{\pi_k}} \left(\frac{l_{(r)} - \mu_{k,(r)}}{\sigma_{k,(r)}} \right) & ;\ d = K + D(k-1) + r, k \in [1, K], r \in [1, D] \\ \frac{\gamma_k(l)}{\sqrt{2\pi_k}} \left(\frac{(l_{(r)} - \mu_{k,(r)})^2}{\sigma^2_{k,(r)}} - 1 \right) & ;\ d = (1+D)K + D(k-1) + r, k \in [1, K], r \in [1, D] \end{cases} \end{aligned} \tag{4.10}$$

With z_{ld} known, local feature relevances $R_l^{(2)}$ and pixel-wise relevances $R_p^{(1)}$ are computed by following the procedures applying to general Bag of Words models outlined in Section 2.4.2. The decomposition process with explicit redistribution formulas is illustrated in Figure 4.1.

4.4 A Measure of Dependency on Image Context for Prediction as a Function of Relevance

In order to quantify the importance of image context to a particular classification task, we define a *outside-inside relevance ratio* metric: The distribution of positive values in a relevance map can be used for assessing the importance of context *for* a particular image classification task. If bounding box annotations are available (as for the Pascal VOC dataset), we can compute a measure as:

$$\mu = \frac{\frac{1}{|P_{out}|} \sum_{q \in P_{out}} R_q^{(1)}}{\frac{1}{|P_{in}|} \sum_{p \in P_{in}} R_p^{(1)}} \tag{4.11}$$

with $|\cdot|$ being the cardinality operator and P_{out} and P_{in} being the set of pixels outside and inside the bounding box, respectively. A high relevance ratio indicates that the classifier uses a lot of context to support the decision. A low relevance ratio indicates that the classifier

TABLE 4.1: **Prediction performance of the trained FV model and DNN in average precision (AP) per class, in percent.**

	aer	**bic**	**bir**	**boa**	**bot**	**bus**	**car**
FV	79.08	66.44	45.90	70.88	27.64	69.67	80.96
DNN	88.08	79.69	80.77	77.20	35.48	72.71	86.30
	cat	**cha**	**cow**	**din**	**dog**	**hor**	**mot**
FV	59.92	51.92	47.60	58.06	42.28	80.45	69.34
DNN	81.10	51.04	61.10	64.62	76.17	81.60	79.33
	per	**pot**	**she**	**sof**	**tra**	**tvm**	**mAP**
FV	85.10	28.62	49.58	49.31	82.71	54.33	59.99
DNN	92.43	49.99	74.04	49.48	87.07	67.08	72.12

focuses instead on the object to support its decision. Note that this measure can not be 100% accurate in most cases, since for example for slim but obliquely angled objects – *e.g.* aeroplanes photographed during lift-off – the typically rectangular bounding box areas which are aligned to the image axes will also cover a considerable amount of image background.

4.5 Experimental Evaluation

All measurements in this chapter are carried out on Pascal VOC 2007 (Everingham et al., 2010) test dataset. Fisher vectors are computed using the encoding evaluation toolkit (version 1.1) from (Chatfield et al., 2011) with settings as in the paper. The Fisher vectors are trained on the training and validation subsets of the Pascal VOC 2007 dataset. The Neural Network model is finetuned on the training and validation subsets of the 2012 version of the Pascal VOC dataset, starting from the BVLC reference classifier of the Caffe package (Jia et al., 2014), pre-trained on ImageNet, with a base learning rate of 0.001 using a multi-label hinge loss. Further details on neural network training and data preprocessing can be found in Appendix C.1.

As we are interested in the ability of a Neural Net to use context, we do not use the bounding box ground truth to extract image patches which cover parts of bounding boxes. Instead we create four corner and one center crop per image together with mirroring, resulting in 10 training patches per image. Test scoring is done in the same fashion. This corresponds to a setting with only a few number of test windows, in which one would use larger patches during training and testing. The region-wise relevance scores are computed for FV as described in Section 4.3 using the ϵ-stabilized decomposition rule (Equation (2.22)) with parameter $\epsilon = 1$ and $\epsilon = 100$. For Neural Networks we use $\alpha\beta$-LRP (Equation (2.26)) with $\alpha = 2, \beta = -1$. The prediction performance of both trained models is reported in Table 4.1 per object class[3].

4.5.1 Are Fisher Explanations Meaningful?

The first step before measuring the Fisher vector model's dependency on contextual information for prediction is to validate whether the computed scores for a pixel or a region are meaningful at all. Figure 4.2 depicts heatmaps computed on exemplary test images of the Pascal VOC data set considering the prediction score for a particular class. The quality of these explanations can be intuitively assessed by a human, *e.g.* it makes perfectly sense that the Fisher vector classifier learned that tires are representative for the class "bike", rail tracks are indicative for the class "train" and tableware is important for classifying images of class "dining table". These examples show that the largest part of the relevance does not necessarily need to collect on the object itself, on the contrary it may be the context which is the informative part.

[3]Class name abbreviations as used in various tables and figures are highlighted in bold font; **aer**oplane, **bic**ycle, **bir**d, **boa**t, **bus**, **car**, **cat**, **cha**ir, **cow**, **din**ingtable, **dog**, **hor**se, **mot**orbike, **per**son, **pot**tedplant, **she**ep, **sof**a, **tra**in, **tvm**onitor

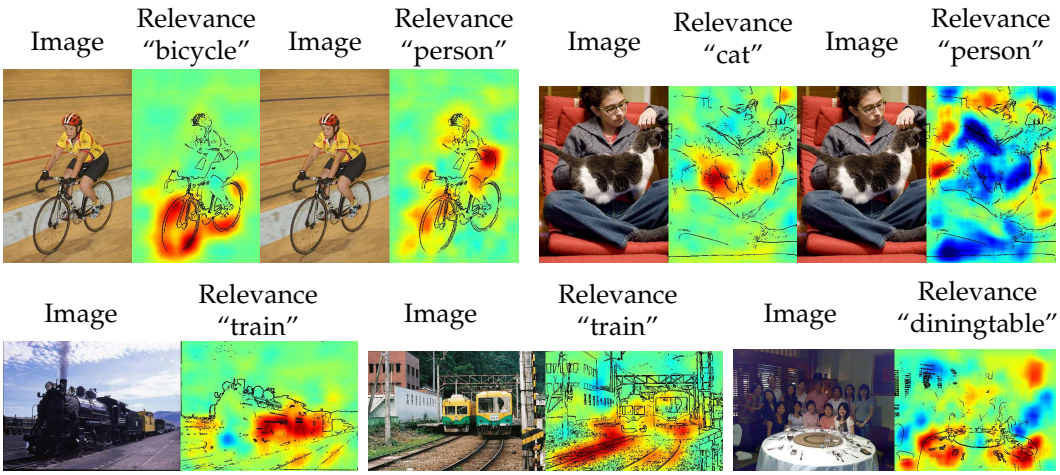

FIGURE 4.2: **Images shown next to the heatmaps computed by application of LRP on the FV model when considering the prediction score for a particular class.**

In order to objectively validate that the Fisher vector heatmaps are meaningful we evaluate the decrease of the prediction score under perturbations. The idea is that a region such as an image patch is highly relevant, if modifying it results for most modifications in a sharp decline of the prediction for the whole image.

This notion of relevant regions can be used for evaluation of region scores by sorting image regions along descending scores. Then, for each region in the sequence the average decrease of predictions is measured. The result is a graph as a function of the sequence index. Thus under this evaluation scheme, a region-wise score performs well if it assigns highest scores to regions which are most sensitive on average under perturbations and yield the sharpest decline of the prediction score. Chapter 3 and (Samek et al., 2017) introduced this setup and evaluated LRP and the methods of (Simonyan et al., 2013; Zeiler et al., 2014) for Deep Neural Networks tested on the ImageNet (Russakovsky et al., 2015), SUN397 (Xiao et al., 2010) and MIT Places (Zhou et al., 2014b) datasets.

Here we show that relevance scores computed with LRP are also meaningful for Fisher vectors. Instead of modifying the input image on pixel level as it is done in Chapter 3, we perturb the set of local descriptors L (computed over regions of the input image) by replacing descriptors l with substitutes l_λ generated via the GMM λ. Considerations regarding the adaption of the perturbation algorithm in Equation (3.6) from Chapter 3 can be found in Appendix C.2. The prediction score is averaged over a number of repetitions of the perturbation experiment, in order to capture the average change of the classifier.

Figure 4.3 shows a comparison of ϵ-LRP (Equation (2.22)) against random orderings, the $\alpha\beta$-decomposition rule (Equation (2.26)) with $\alpha = 2, \beta = -1$ and a decomposition which only relies on the magnitude of the mappings $|z_{ij}|$ (and not the sign) as in the counter example given in Equation (2.7).

Fisher vector mappings compute mappings of both positive and negative sign, which both can contribute meaningfully as evidence for an object category. It is therefore intuitive that the ϵ-stabilized decomposition approach performs best by a large margin. The $\alpha\beta$-rule performs well for DNNs with ReLU activation units, since here activated neurons always fire positively and thus activate succeeding neurons via positive weights. Negatively weighted connections do have an inhibitory effect. Weighting activating forward messages $x_i w_{ij}^+$ and inhibiting forward messages $x_i w_{ij}^-$ separately with α and β reflects that paradigm well. Such a behaviour is not given for Fisher vectors, where negative mappings can not be related to a inhibition of prediction scores. The results from Figure 4.3 motivate the application of ϵ-LRP with $\epsilon = 100$ to measure the importance of context used for prediction. The choice of a larger value for ϵ suppresses noisy relevance attributions caused by the computation of proportional weightings $v_{ij} = \frac{z_{ij}}{z_j}$ with very small z_j, causing numerically unbounded v_{ij}.

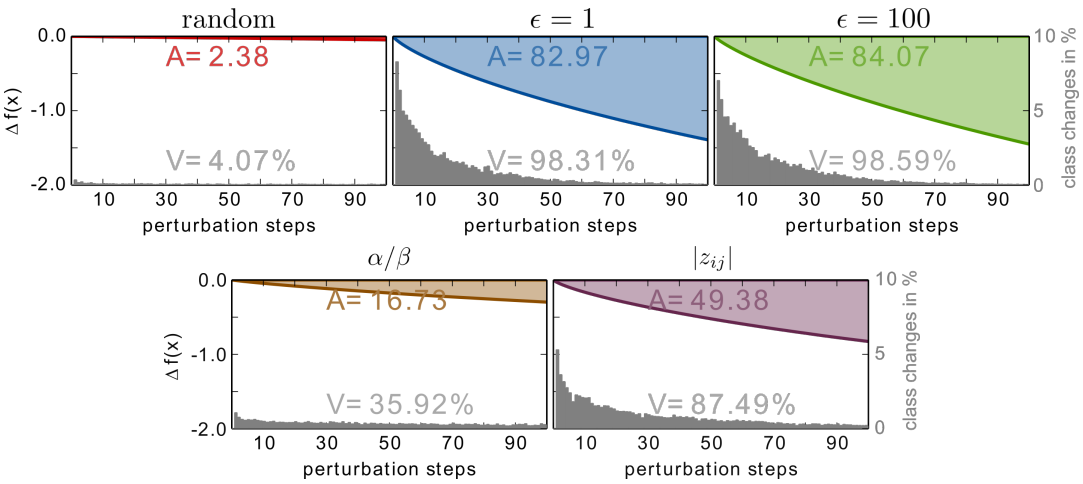

FIGURE 4.3: **Heatmap quality measurements for Fisher vectors and different decomposition rules.** The value A measures the area above the curve between the original prediction $f(\mathbf{x})$ and the averaged perturbed prediction at step i in the sequence of regions. V represents ratio of perturbed samples being predicted differently due to perturbations after 100 perturbation steps. The histogram in gray colour represent the ratio of input samples with changes in predicted class per perturbation step.

4.5.2 Investigating Shallow vs. Deep Features

We investigate in the light of the LRP framework the differences of strategies used to classify images between (1) a shallow model operating on high-resolution images; the FV model, and (2) a deep model operating on lower-resolution images; the DNN model. We consider first the class "sheep" for which the DNN produces much better predictions than the FV model (25% superior average precision in absolute terms according to Table 4.1). Example of two images of class "sheep" and the corresponding heatmaps for the FV and DNN models are shown in Figure 4.4.

The LRP analysis reveals that the FV and DNN models use clearly different strategies to predict the class: The FV model bases its decision on the wool texture typical of the sheep and available at high-resolution, but ignores the exact shape of the sheep. Interestingly, relevance is also allocated to the context (here, positive relevance for the grass and negative relevance for the human face), indicating that the context is an essential component of the classifier and modulates the prediction score positively or negatively.

On the other hand, the DNN assigns a large proportion of positive relevance to the border of the sheep, thus, showing that the shape of the sheep (*e.g.* its contour) is exploited in order to improve the prediction. Furthermore, for the DNN, the LRP method does not assign strong relevance to contextual elements such as the grass, or the human face, nor to the wool texture of the sheep, which is harder to detect due to the low resolution of images provided to the DNN: The FV model computes local features from $(480 \times X)$ or $(X \times 480)$-sized images, while the inputs for the DNN are first padded to a square shape (by repeating pixel values from the image border), and then scaled to (256×256) pixels from which the (227×227)-sized center is cropped.

Overall, the LRP analysis indicates that the far superior prediction performance of the DNN model must be attributed in largest part to the ability to model the exact shape of the sheep, making all remaining contextual or texture features less relevant. On the other hand, the less accurate FV model does benefit from the weak correlations between object class, texture and context to improve prediction quality.

4.5.3 Test Error and Model Quality

For other classes, it can be observed in Table 4.1 that test error of the FV model is almost on par with the one of the DNN. We investigate whether high test performance is predictive

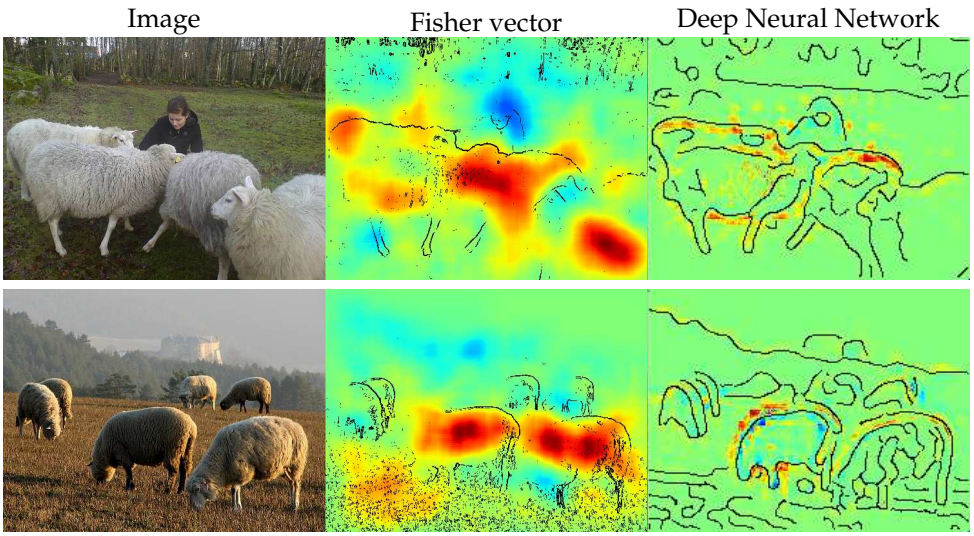

FIGURE 4.4: **Images of the class "sheep", processed by the FV and DNN models and LRP.**

of the ability of the model to extract meaningful features for a given class, or whether the decision is based mostly on undesirable contextual or artefactual features.

Contextual Features

As an illustrative example, we consider the class "boat", where the performance of the DNN superior by less than 7% in absolute terms to the FV model[4]. It is tempting to conclude that, for the class "boat", both models should have learned a set of features of similarly high quality. LRP analysis gives a different answer: Figure 4.5 (left) shows the relevance maps produced by the FV and DNN models on two archetypical images of the class "boat".

For the DNN, LRP assigns most of the relevance to pixels corresponding to visible boat-like structures such as the bridge and sails. On the other hand, for the FV model, LRP assigns most relevance to the water below the boat, *i.e.* the FV model does not recognize the object itself, but its context. The distribution of pixel-wise relevance averaged over relevance maps – computed over all landscape-format images of the class "boat" – corroborates what was observed for two selected images, in particular, a focus of the FV model on the bottom part of the image where water can be expected, and a focus of the DNN model on the middle part of the images centered on the photographed boats. We can conclude from our analysis using LRP, that while both classifiers have a roughly similar level of average precision on the test images with class "boat", FV's performance is likely to decrease drastically if one were to consider boats located outside the water as test images. On the other hand, performance of the DNN would be less affected. Therefore, test error is a superficial predictor of model quality in this case.

Artefactual Features

A second example where high accuracy does not necessarily translate into high quality features is for the class "horse". This class is predicted with similar performance by the FV and DNN models ($\approx$ 1% difference in average precision).

Figure 4.5 (right) shows relevance maps for the FV and DNN model and an image of a horse with rider. While the DNN assigns relevance on the actually shown "horse", the FV assigns almost all relevance in the bottom-left corner the image, where careful inspection of the image reveals the presence of a copyright watermark. Thus, the decision of the FV model is in large parts based on the presence of the copyright tag, which is discriminative of the class horse. Removing the copyright tag completely changes the FV relevance map in the

[4]Note that for other classes such as "sheep" or "bird", the DNN performance is superior by 25% or more.

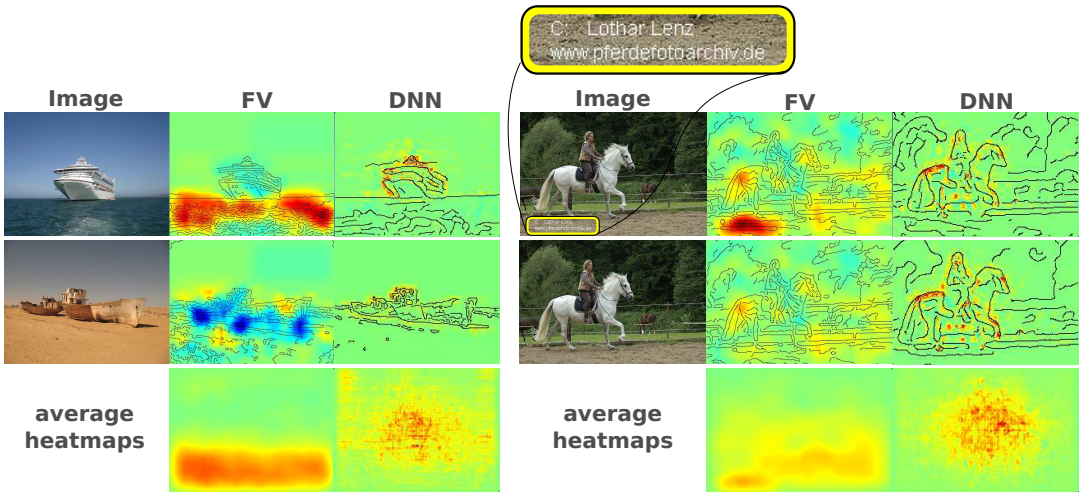

FIGURE 4.5: **Relevance maps for classes "boat" and "horse".** *Top and center*: Images of the classes "boat" and "horse", processed by the FV and DNN models and heatmapped using LRP. *Bottom*: Average relevance maps over a random sample (of size between 47 and 177) of the distribution for each class and model. On the second image of class "horse", the copyright tag has been removed.

bottom left covered by the watermark (and the predictor output for class "horse"), while the relevance map and model prediction for the DNN remain almost unchanged.

If the copyright tag is removed, the DNN is still able to predict the image because the pixels that support its decision are not affected. On the other hand, FV model prediction quality will be considerably reduced. The systematic focus of the FV model on the copyright tag is confirmed in the average relevance map in Figure 4.5, where the bottom-left corner is assigned large amount of positive relevance. Therefore, for this class again, test error does not predict well model quality.

4.5.4 Quantitative Analysis of Context Dependence

While we have so far provided a qualitative interpretation of FV relevance maps for examples and classes of interest, we can more systematically measure whether the model uses context or the actual object, by measuring for each classes and models the outside-inside relevance ratio μ computed by Equation (4.11). The resulting measurements per object class are shown in Figure 4.6, where higher values signify a higher dependency on contextual image features. Generally, the FV model uses more context than the DNN, as evident via the higher measured relevance ratios. However, there are significant differences between classes: Classes where the use of context by the FV model is particularly high are "boat" and "airplane", the first of which we have studied qualitatively in the previous section. For these two respective classes, the water and the sky are important contextual elements that support the decision of the Fisher vector model, due to their strong correlation.

For other classes such as "bicycle", "car", "motorbike", or "sheep", the Fisher vector model does not use much context. For the first three classes, the urban environment surrounding these classes is not predictive of the object being detected, *i.e.* it could not discriminate between these three classes based on the context only. For the last class, as it has been discussed in Section 4.5.2, the wool texture of the sheep (which lies inside the sheep bounding box) is a reasonable predictor for the class "sheep", although the actual object sheep (*i.e.* defined by its shape or contour) is not being used in its full extent.

As for DNNs, classes with least context usage are "aeroplane", "bird", "sheep", "dog", "car", "cat" and "tvmonitor". Each of those is associated with a significantly better score achieved by the DNN. Another group of classes with high importance of image context for both the Fisher model and DNN are "chair", "diningtable", "pottedplant" and "sofa" which share a semantic of indoor room sceneries.

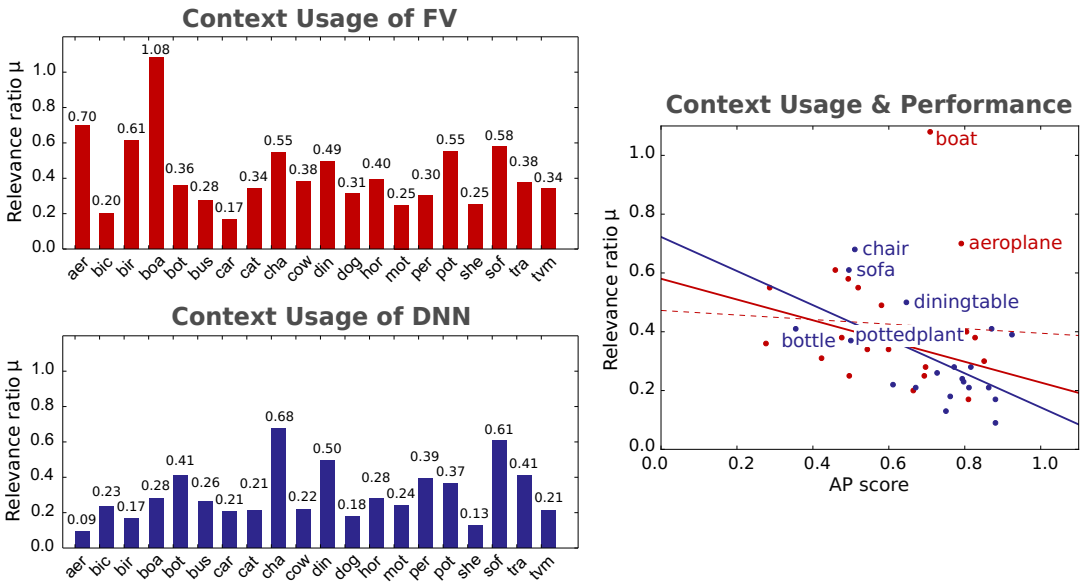

FIGURE 4.6: **Importance of context for prediction per model and class.** *Left*: Measurements reflecting the importance of image context μ per class and model. Higher values correspond to on average large amounts of positive relevance being located outside the object bounding boxes. *Right*: Red dots show the relationship between AP score and importance of context per class for the FV model. Blue dots show the same results for the DNN. The red solid line tracks the linear relationship between AP score and importance of context μ for the FV model when ignoring the outliers (classes "aeroplane" and "boat"). The dashed red line shows the trend for all 20 classes. The blue line shows the trend for all classes and the DNN model. The named blue dots are classes for which the DNN model uses relatively high amount of context information due to feature sharing, caused by exceptionally high co-occurrence rates in the class labels (Figure 4.7).

We attribute the dependency on contextual image information to a combination of circumstances: For one, the FV mapping step includes the subdivision of the input image into multiple image sub-areas, a technique known to considerably boost the predictive performance of Bag of Words models due to the incorporation of weak geometric information. In this mapping scheme, one FV representation is computed for each image sub-area, spanning a separate set of dimensions in the final image representation in vector form. This encourages the classifier to optimize by concentrating on the subset of input dimensions providing information which correlates most with the expected label. Secondly, the Pascal VOC 2007 training dataset is comparatively small and lacks diversity. Some classes are under-represented or class labels frequently co-occur. The lack in diversity is common cause for the development of prediction biases, while object co-occurrence will cause the model to also learn features from other classes, located outside the target class object's bounding box area.

The DNN predicts dominantly based on the objects themselves. Therefore, most class context values reported in Figure 4.6 (left) are much lower for the DNN than for the FV model. The DNN has been fine-tuned on Pascal VOC data and moreover many of the 1000 classes from the ImageNet challenge semantically overlap with the classes present in the Pascal VOC data. The class "horse" for example corresponds to the (semantic) subcategories "zebra" and "sorrel" in the ImageNet label set. While Pascal VOC provides the class "cat", ImageNet has a range of labels describing subtypes of cats ("tabby", "burmese cat", "manx", ...). This visually and semantically similar overlap allows the network to resort to robust prior knowledge accumulated during training for the comparatively larger and more diverse ImageNet. This prior knowledge – the learned set of filters insensitive to the biases, but sensitive to the appearance of the objects in the Pascal VOC data – benefits the fine-tuning and results in a model which – loosely speaking – merely has to adapt its outputs to the new semantic feature groupings in the Pascal VOC label set.

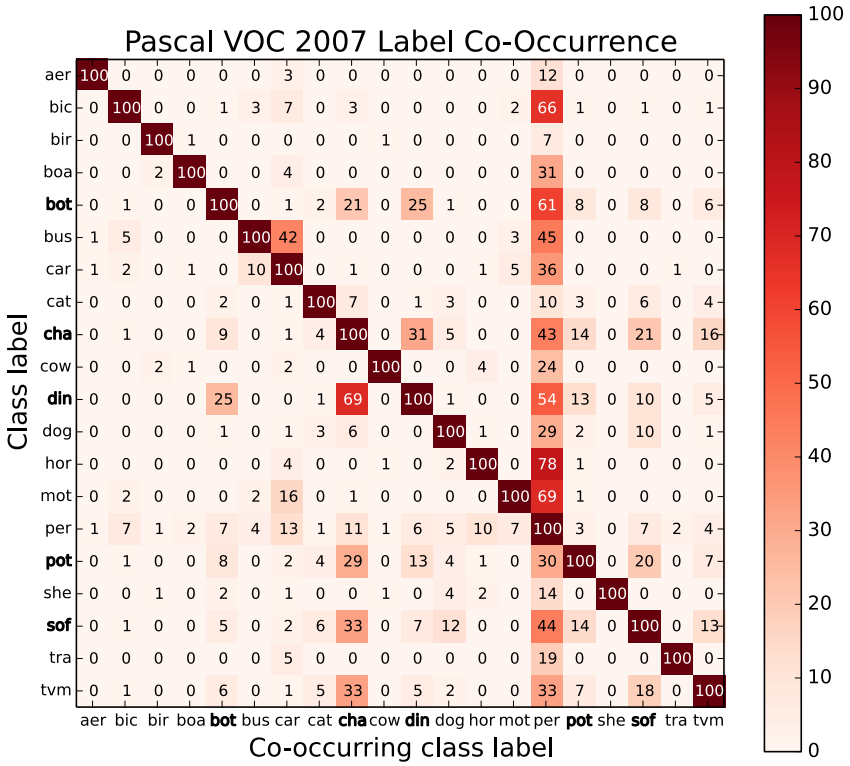

FIGURE 4.7: **Pascal VOC 2007 label co-occurrence rates.** Values are in percent. The entries visualize the rate at which the classes at the columns co-appear in samples belonging to classes indexing the rows of the matrix. Most notably is the frequent appearance of class "person", which *e.g.* appears in 66% of "bicycle" samples. Also the *living room* classes ("bottle", "chair", "diningtable", "pottedplant", "sofa") often appear in the same images, explaining the high use of contextual information reported in Figure 4.6.

There are, however, outlying classes in the measurements shown in Figure 4.6 (left) for the DNN model, namely the classes "chair", "diningtable" and "sofa" and other classes describing interior items. This set of labels often co-occurs in the images of the Pascal datasets, showing cluttered indoor scenes (see Figure 4.7). The frequently and simultaneously present set of features makes it difficult for both models to distinguish between the present classes as individual categories. Furthermore, from the results in Figure 4.6 (left) and Table 4.1 we may conjecture that performance may indeed correlate with "object understanding", as shown in Figure 4.6 (right) which combines both results.

4.5.5 Comparing Network Depth

Our results on ILSVCR 2012 validation data show that, when performing LRP on the BVLC CaffeNet (Jia et al., 2014) versus GoogleNet (Szegedy et al., 2015) and the VGG CNN S (Chatfield et al., 2014), the use of contextual information is much lower for the deeper and better performing GoogleNet[5]. All models have been used as-is, *i.e.* no changes have been made to the architecture or model weights as available via the Caffe Model Zoo[6].

In addition to depth, the type of layers (*e.g.* inception, normalization) may have an impact on the sparsity and should be subject to further studies. Figures 4.8 and 4.9 present the results quantitatively and as exemplary relevance maps for the CaffeNet (Jia et al., 2014) and VGG CNN S (Chatfield et al., 2014) – which has slightly lower error rate than the former – and GoogleNet (Szegedy et al., 2015). The latter two use smaller kernels with less stride at lowest level compared to the CaffeNet. We noted on many examples, that GoogleNet is

[5]Top-5 test error rates on ILSVCR 2012: CaffeNet 16.4%, VGG CNN S 13.1%, GoogleNet: 6.7%

[6]https://github.com/BVLC/caffe/wiki/Model-Zoo

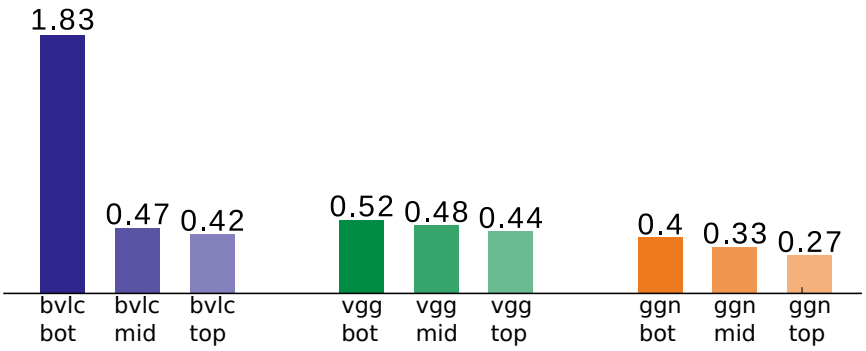

FIGURE 4.8: **DNN context scores for ImageNet 2012 for the BVLC CaffeNet (bvlc, blue), VGG CNN S (vgg, green) and GoogleNet (ggn, orange)**. *bot* are 333 classes with lowest prediction accuracy wrt the used network and *top* are 333 classes with highest prediction accuracy. GoogleNet uses less context. As for an explanation for the higher values of context importance relative to the results on PASCAL VOC 2007, visual inspection revealed that many of the ImageNet bounding boxes cover much less of the object than those used in Pascal VOC.

much sparser than the other two and tends to ignore irrelevant edges, for example its reaction to the gradient between the dark green trees and the sky in the motorscooter example picture is the weakest of all three nets.

4.6 Limitations

The relevance maps shown in this chapter point out a problem for the qualitative comparison of computed relevance maps between different models. While the observed effect might not be as obvious between Neural Networks, it becomes rather apparent when comparing the DNN and FV Model: The granularity of the computed relevance decomposition may differ considerably, *e.g.* as observable in Figure 4.4, and Figure 4.9 to a lesser extent. This effect is tied to model architecture, *i.e.* in the size, scale and stride of (filter) mappings in the forward pass, and whether they are decomposable at all via the availability of mapping quantities z_{ij} (which are not always given for Bag of Words models). While in some cases relevance decompositions may be adapted to obtain relevance maps of corresponding scale (see Chapter 5), no suitable solution may be available in others, as with the Bag of Words models decomposed in Chapter 2, Section 2.4.3. The difference in relevance map granularity between models only has a minor impact to our quantitative analyses regarding the use of image content for prediction, due to the property of meaningfully aggregatability of relevance over regions of the input image (see Section 3.3.5).

4.7 Conclusion

In this chapter, we have so far analyzed what make Fisher vector models and Deep Neural Networks (DNN) decide for a particular class. To achieve this, we have employed LRP to determine which pixels of an input image are used by a classifier to support its decision. LRP was extended to Fisher vector models, and validated using input perturbations from Chapter 3. Our novel comparative analysis of FV and DNN classifiers corroborates empirically previous intuition relating the architecture of the classifier to the features it is able to extract. In particular, our analysis shows that the FV model compensates its lack of depth and prior knowledge from pretraining by the use of contextual information – potentially artefacts – that are weakly correlated to the object class. We thus demonstrate that the generalization capability of Fisher vector models can be overstated if test images also include similar context. On the other hand, DNNs base their decision on the actual object

to detect and largely ignores its context. This focus on object detection has to be attributed to the higher overall predictive accuracy of the model, that removes the need for contextual information – even if the latter is discriminative. The focus on detection must also be attributed to the deep multitask properties of the DNN that favors composition of natural image features over lower-level features such as copyright text. These results argue in favor of incorporating saliency techniques into the data collection and model selection processes. The interpretable visual feedback that relevance maps provide can be used in particular to verify that the considered classifier bases its decision on the right set of features, and in the contrary case, select another model, or extend the dataset in a way that artefactual features can no longer support the classification decision.

FIGURE 4.9: **Comparison of relevance maps for different pretrained models on ImageNet for classes "scooter", "frog", "cat", "taxi", "wolf", "palace", and "sombrero".** From left to right: Input, relevance maps for the BVLC CaffeNet, VGG CNN S and GoogleNet. Relevance maps for GoogleNet are particularly sparse which holds for many other examples.

Chapter 5

Investigating Pretraining and Preprocessing on DNNs for Face Categorization

Chapter 4 has taught that there is a connection between the complexity of a predictor, its performance and its ability to focus on the (important bits in the) input. Results comparing different Neural Network architectures operating on the 1000 classes of ImageNet further confirm this observation.

Another field Deep Neural Networks recently excelled in is the recognition of age and gender from human face images. Again, it is not clear which facial features are actually used for prediction and how these features depend on image preprocessing, model initialization and architecture choice. We take the application domain of face recognition as an opportunity to deepen our investigations on Deep Neural Network predictions to address the effects of these factors.

In detail, our work compares four popular Neural Network architectures, studies the effect of pretraining, evaluates the robustness of the considered alignment preprocessings via cross-method test set swapping and intuitively visualizes the model's prediction strategies in given preprocessing conditions using Layer-wise Relevance Propagation. Our evaluations on the challenging Adience benchmark show that suitable parameter initialization leads to a holistic perception of the input by the trained models, compensating artefactual data representations. With a combination of simple preprocessing steps, we reach state of the art performance in gender recognition on the challenging Adience benchmark dataset (Eidinger et al., 2014). This section covers the contributions from (S. Bach et al., 2016) and (Lapuschkin et al., 2017).

5.1 Introduction

Automated facial recognition and estimation of gender and age using machine learning models has held a high level of attention for more than two decades (Kwon et al., 1994; O'toole et al., 1997; Baluja et al., 2007; Guodong et al., 2009; Gao et al., 2009) and has become ever more relevant due to the abundance of face images on the web, and especially on social media platforms. The introduction of DNN models to this domain has largely replaced the need for hand crafted facial descriptors and data preprocessing, while simultaneously increasing possible prediction performances at an incredible rate. DNN models have been not only successfully applied for age and gender recognition, but also for the classification of emotional states (Arbabzadah et al., 2016). In the previous four years alone, age recognition rates increased from 45.1% (Eidinger et al., 2014) to 64% (Rothe et al., 2016) and gender recognition rates from 77.8% to reportedly 91% (Dehghan et al., 2017) on the recent and challenging Adience benchmark (Eidinger et al., 2014), mirroring the overall progress on other available benchmarks such as the Images of Groups data set (Gallagher et al., 2009), the LFW data set (G. B. Huang et al., 2008) or the Ghallagher Collection Person data set (Gallagher et al., 2008).

In this chapter,

(1) we compare the influence of model initialization with weights pretrained on two real

world data sets to random initialization and analyze the impact of (artefactual) image preprocessing steps to model performance on the Adience benchmark dataset for different recent DNN architectures.
(2) we can show that suitable pretraining can yield a robust set of starting model weights, compensating artefactual representation of the data, via cross-method test set swapping.
(3) using Layer-wise Relevance Propagation (Chapter 2, Section 2.3), we visualize on pixel level, how those choices made prior to training affect the classifier interacts with the input, *i.e.* how the provided input is used to make a decision, and what parts of it in which manner.
(4) in order to obtain comparable results on the same local scale between models, we introduce the $\flat$-decomposition rule for LRP in an application setting and present a refined parameterization of LRP decomposition rules for sequential DNN architectures.
(5) finally, we rectified the performance of (Rothe et al., 2016) on gender recognition referred to in (Dehghan et al., 2017) with a more likely result and report our own result, slightly exceeding that baseline:
(6) Via a combination of simple preprocessing steps, we can reach state of the art performance on gender recognition from human face images on the Adience benchmark dataset.

5.2 Related Work

One of the more recent face image data sets is the Adience benchmark (Eidinger et al., 2014), which has been published in 2014, containing 26,580 photos across 2,284 subjects with a binary gender label and one label from eight different age groups[1], partitioned into five splits. The key principle of the data set is to capture the images as close to real world conditions as possible, including all variations in appearance, pose, lighting condition and image quality, to name a few. These conditions provide for an unconstrained and challenging learning problem: The first results on the Adience benchmark achieved 45.1% accuracy for age classification and 77.8% accuracy for gender classification using a pipeline including a robust, (un)certainty based in-plane facial alignment step, Local Binary Pattern (LBP) descriptors, Four Patch LBP descriptors and a dropout-SVM classifier (Eidinger et al., 2014). For reference, the same classification pipeline achieves 66.6% accuracy for age classification and 88.6% accuracy for gender classification on the Ghallagher data set. The authors of (Hassner et al., 2015) introduce a 3D landmark-based alignment preprocessing step, which computes frontalized versions of the unconstrained face images from (Eidinger et al., 2014), which slightly increases gender classification accuracy to 79.3% on the Adience data set, otherwise using the same classification pipeline from (Eidinger et al., 2014).

The first time a DNN model was trained on the Adience benchmark was by (Levi et al., 2015). The authors did resort to an end-to-end training regime, *e.g.* the face frontalization preprocessing from (Hassner et al., 2015) was omitted and the model was completely trained from scratch, in order to demonstrate the feature learning capabilities of the Neural Network type classifier for face image data. The architecture used in (Levi et al., 2015) is very similar to the BVLC Caffe Reference Model (Jia et al., 2014), with the fourth and fifth convolution layers being removed. The best reported accuracy ratings increased to 50.7% for age classification and 86.6% for gender classification, using an over-sampling prediction scheme with 10 crops taken from a sample (4 crops from the corners and the center crop, plus mirrored versions) instead of only the sample by itself (Levi et al., 2015).

To the best of our knowledge, the current state of the art results for age and gender predictions are reported in (Rothe et al., 2016) and (Dehghan et al., 2017) with 64% and 91% accuracy respectively. The model from (Rothe et al., 2016) was the winner of the ChaLearn Looking at People 2015 challenge (Escalera et al., 2015) and uses the VGG-16 layer architecture (Simonyan et al., 2014), which has been pretrained on the IMDB-WIKI face data set. This data set was also introduced in (Rothe et al., 2016) and collects 523,051 labelled face images collected from Internet Movie Data base (IMDb) and Wikipedia. Prior to pretraining on the IMDB-WIKI data, the model was initialized with the weights learned for the ImageNet 2014 challenge (Russakovsky et al., 2015). The authors attribute the success of their model to large amounts of (pre)training data, a simple yet robust face alignment preprocessing step (rotation only), and an appropriate choice of network architecture.

[1] In years: (0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60+)

TABLE 5.1: **An overview over the developments for age and gender recognition results on the Adience benchmark in recent years.** Accuracy values are reported in percent. Our best result for gender recognition is highlighted with bold font, while the same configuration applied to age categorization did not improve above related work and is shown in gray colour. Dashes are unavailable results.

	gender	age	age (1-off)
Eidinger et al., 2014	77.8	45.1	79.5
Hassner et al., 2015	79.3	–	–
Levi et al., 2015	86.8	50.7	84.7
Rothe et al., 2016	–	64.0	96.6
Dehghan et al., 2017	91.0	61.3	–
Ours	**92.7**	63.0	96.0

The 91% accuracy achieved by the commercial system from (Dehghan et al., 2017) is supposedly backed by 4,000,000 carefully labelled but non-public training images. The authors identify their use of landmark-based facial alignment preprocessing as a critical factor to achieve the reported results, despite recent studies reporting contradicting findings (F. Chang et al., 2017) and reflecting our own results. Unfortunately no details are given about the model architecture in use. The authors of (Dehghan et al., 2017) compare their results to (Rothe et al., 2016) and other systems, yet only selectively list the age estimation performances of competing methods, such as (Rothe et al., 2016). The authors of (Dehghan et al., 2017) also report the gender recognition performance of (Rothe et al., 2016) as only 88.75%, which is rather low given the early results from (Levi et al., 2015), the performance of (Rothe et al., 2016) on age recognition and our own attempts to replicate the models of referenced studies.

Recapitulating, we can identify three major factors contributing to the performance improvements among the models listed in Table 5.1: (1) Changes in architecture. (2) Prior knowledge via pretraining. (3) Optional dataset preparation via alignment preprocessing.

In the following, this chapter will briefly describe a selection of DNN architectures and investigate the influence of random weight initialization against pretraining on generic (ImageNet) or task-specific (IMDB-WIKI) real world data sets, as well as the impact of data preprocessing by comparing affine reference frame based alignment techniques to coarse rotation-based alignment. Due to its size and the unconstrained nature of the data and the availability of previous results, we use the Adience benchmark data set as an evaluation sandbox. The dataset is available as a rotation aligned version, and as a version with images preprocessed using the affine in-plane alignment (Eidinger et al., 2014), putting the shown faces closer to a reference frame of facial features. We then use Layer-wise Relevance Propagation to give a glimpse into the model's prediction strategy, visualizing the facial features used for prediction on a per-sample basis in order to explain major performance differences.

5.3 Architectures, Preprocessing and Model Initialization

This section provides an overview about the evaluated DNN architectures, data preprocessing techniques and weight initialization choices. All models are trained using the Caffe Deep Learning Framework (Jia et al., 2014), with code based on the configurations to reproduce the results from (Levi et al., 2015) as available on github[2].

5.3.1 Evaluated Models

We compare the architectures of the model used in (Levi et al., 2015) (in the following referred to as AdienceNet), the BVLC Caffe Reference Model (Jia et al., 2014) (alias: CaffeNet), the GoogleNet (Szegedy et al., 2015) and the VGG-16 (Simonyan et al., 2014), on which state

[2] https://github.com/GilLevi/AgeGenderDeepLearning

of the art performance on age classification has been reported in (Rothe et al., 2016). The AdienceNet is structurally similar to the CaffeNet, with the main difference being smaller convolution masks learned in the input layer ((7×7) vs (11×11)) and two convolution layers fewer being used. The number of hidden units composing the fully connected layers preceding the output layer is considerably lower (512 vs 4096) for AdienceNet. The VGG-16 consists of 13 convolution type layers of very small kernel sizes of 2 and 3, which are interleaved with similarly small pooling operations, followed by two fully connected layers with 4096 hidden units each, and a fully connected output layer. The fourth model we use and evaluate is the GoogleNet, which connects a series of inception layers. Each inception layer realizes multiple convolution/pooling sequences of different kernel sizes (sizes (3×3) to (7×7) in the input inception module) in parallel, feeding from the same input tensor, of which the outputs are then concatenated along the channel axis. Compared to the VGG-16 architecture, the GoogleNet is fast to train and evaluate, while slightly outperforming the VGG-16 model on the ImageNet 2014 Challenge with 6.6% vs 7.3% top-5 error in the classification task (Russakovsky et al., 2015).

5.3.2 Data Preprocessing

One choice to be made for training and classification is regarding data preprocessing. The SVM-based system from (Hassner et al., 2015) improves upon (Eidinger et al., 2014) by introducing a 3D face frontalization preprocessing step, with the goal of rendering the inputs to the pipeline invariant to changes in pose. Landmark-based preprocessing also is identified in (Dehghan et al., 2017) as an important step for obtaining the reported model performances. Both (Levi et al., 2015) and (Rothe et al., 2016) only employ simple rotation based preprocessing, which roughly aligns the input faces horizontally, trusting the learning capabilities of Neural Networks to profit from the increased variation in the data and learn suitable data representations.

The Adience benchmark data set provides both a version of the data set with images roughly rotated to horizontally aligned faces, as well as an affine 2D in-plane aligned version for download. We prepare training and test sets from both versions using and adapting the original splits and data preprocessing code for (Levi et al., 2015) available for download on github. We also create a mixed data set from a union of both previous data sets, which has double the number of training samples and allows the models to be trained on both provided alignment techniques simultaneously.

5.3.3 Weight Initialization

An invaluable benefit of DNN architectures is the option to use pretrained models as a starting point for further training. Compared to random weight initialization, using a pretrained models as starting points often results in faster convergence and overall better model results, due to initializing the model with meaningful filters, given the pre-training data and the problem at hand's data are relatable.

Here, we compare models initialized with random weights to models starting with weights pre-trained on other (related) data sets, namely the ImageNet data set and the IMDB-WIKI data sets, whenever model weights are readily available. That is, we try to replicate the results from (Levi et al., 2015) and train an AdienceNet model from scratch only, since no weights for either pretraining data set are available. Instead, we use the comparable CaffeNet to estimate the results obtainable when initialzing the model with ImageNet weights. We also train the GoogleNet from scratch and initialized with ImageNet weights. Due to the excessive training time required for the VGG-16 model, we only try to replicate the results from (Rothe et al., 2016) and train models initialized with available ImageNet, and IMDB-WIKI weights.

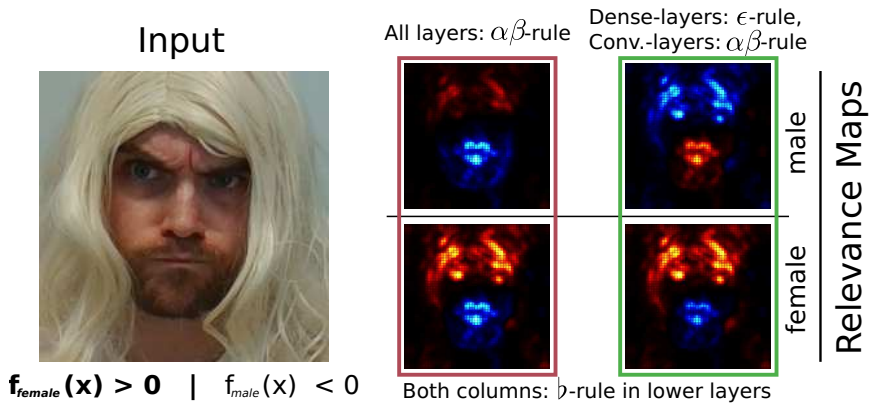

FIGURE 5.1: **An application of the ϵ-rule to the top dense layers of the DNN better represents the model output in terms of relevance decomposition.** *Left:* The given input is predicted as "female" with a negative logit output for the class "male". *Center:* Due to the positivity assumption of the αβ-rule (Montavon et al., 2018), a relevance decomposition using the αβ-rule for all layers results in positive relevances for female features and negative relevances for male features for both classes. *Right:* An application of the ϵ-rule to the final dense layers of the (GoogleNet) model correctly decomposes the negatively signed output for class "male" (Kohlbrenner, 2017). The effect and purpose of the ♭-rule applied to the lowest layers of the model are described in Section 5.4.2.

5.4 Visualizing Model Perception Comparably across Different Architectures and Network Depths

We complement our quantitative analysis in Section 5.5 with qualitative insights on the perception and reasoning of the models by explaining the predictions made via LRP. For the purpose of comparability of all evaluated models, we extend LRP by proposing a novel configuration for decomposing the decision function of sequential, ReLU-activated DNNs.

5.4.1 Refining LRP for Sequential and ReLU-Activated DNNs

The relevance redistribution obtained from Equations (2.9) to (2.13) in Chapter 2, Section 2.2 is a very general one, with exact definitions depending on a neuron or input's type and position in the pipeline.

All DNN models considered in this chapter consist in one part of ReLU-activated (convolutional) feature extraction layers towards the bottom, followed by inner product layers serving as classifiers (Montavon et al., 2010). We therefore apply to inner product layers the ϵ-decomposition rule (Equation (2.22)) with a small $\epsilon = 0.01$ added to the denominator for numeric stability, to truthfully represent the decisions made via the layers' linear mappings consistently.

Since the ReLU activations of the convolutional layers below the fully connected layers serve as a gate to filter out weak activations, we apply the αβ-decomposition formula (Equation (2.26)) with $\alpha = 2, \beta = -1$, which handles the activating and inhibiting parts of the convolution layers' forward mappings z_{ij} separately.

The αβ-rule tacitly assumes that activating outputs – and thus relevance values – of a (final) layer to be positively valued (Montavon et al., 2018). Once this assumption is not fulfilled[3], the sign of the pixel-wise relevance maps will become inverted, due to the initial relevance values $R_k^{(L)} = f_k(\mathbf{x})$ at output layer L for class of interest k disagreeing with that assumption. The work of (Kohlbrenner, 2017) has shown that an application of the

[3] *E.g.* in problem settings such as gender recognition, where the few available classes directly contradict each other: A positive outcome for class "female" will most likely result in a negative outcome for class "male".

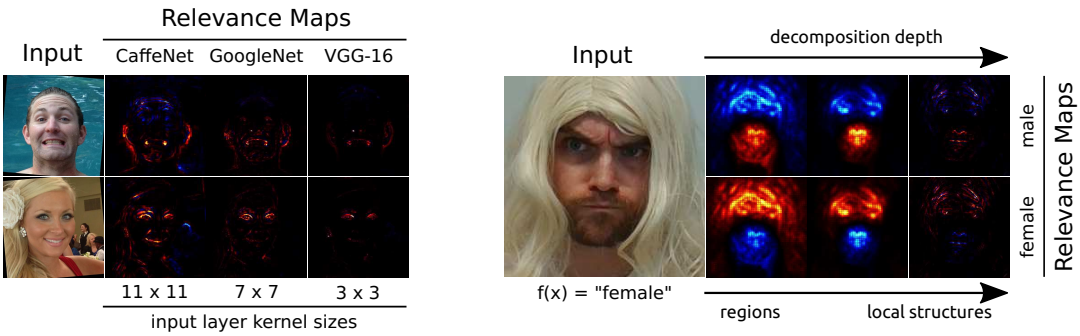

FIGURE 5.2: **The receptive fields of explained neurons dictate the semantics of the resulting relevance maps.** *Left:* Relevance maps down to pixel level for different DNN architectures with different filter sizes given the same inputs. Larger convolution filter sizes in the network's bottom layers correspond to attribution of relevance to larger structures. *Right:* Relevance maps for both gender outputs at different decomposition depth. Decomposition depth affects the receptive fields of the last decomposed neurons and thus the semantics of the resulting relevance maps. Relevance decomposition further down to pixel level promote relevance allocation to finer structures. Earlier application of the $\flat$-decomposition rule (as in "flat" or uniform distribution) leads to larger receptive fields of the last decomposed neurons and thus allocation and semantics of relevance scores corresponding to larger regions. The reasoning of the GoogleNet model when predicting for either of the two gender classes for the given input is antithetic.

ϵ-decomposition rule to fully connected layers and the $\alpha\beta$-decomposition rule to convolutional layers in sequential Neural Networks with ReLU activation units – as it is the case for all evaluated models in this section – yields best results both qualitatitvely and quantitatively. Figure 5.1 demonstrates the described effect and the improved rule parameterization from (Kohlbrenner, 2017) at hand of a gender recognition task. Theoretical insights into above decomposition types can be found in (Montavon et al., 2016; Montavon et al., 2017; Montavon et al., 2018).

5.4.2 Adapting the Decomposition Depth of LRP

All compared DNN models use vastly different convolution filter sizes (from 3 to 11) in the bottom layers, which translates to the output neurons of each DNN's input layer capturing information at different receptive field sizes. Decomposing each of the DNNs layers down to pixel level would thus result in relevance maps of different levels of granularity, focussed on structures of different scale and size. Figure 5.2 (left) shows relevance maps computed as described in Section 5.4.1 until the pixel level is reached for the CaffeNet, the GoogleNet and the VGG-16 model given the same input images. The kernel sizes of the convolutions operating on the networks' input layers are (11×11) with a stride of 4, (7×7) with a stride of 2 and (3×3) with stride of 1 respectively. For the CaffeNet, the application of LRP results in a highlighting of larger facial structures due to the model's filter sizes in the first convolution layer. For the VGG-16 network, the identified relevant image regions – *e.g.* the male person's nostrils or the female person's upper eyelid in contrast to the ears and the eyes plus eyebrows for the CaffeNet – are of a much smaller scale.

In order to compare the DNN models, we want to ensure that the relevance maps for all models correspond to the same local granularity and structural scale to ensure comparability of the prediction behaviour between all architectures at the same semantic level. For that purpose, we construct a decomposition rule we call the $\flat$-decomposition rule, which as been briefly introduced in Chapter 2 as Equation (2.27). Given an upstream neuron j and an

associated relevance score $R_j^{(l+1)}$, we compute downstream relevances as

$$\begin{aligned} \forall i,j: \; z_{ij} &= 1 \\ z_j &= \sum_i z_{ij} \\ R_{i\leftarrow j}^{(l,l+1)} &= \frac{z_{ij}}{z_j} R_j^{(l+1)}, \end{aligned} \tag{5.1}$$

i.e. by setting all $z_{ij} = 1$ we uniformly distribute the upstream relevance value $R_j^{(l+1)}$ of neuron j across all its input neurons. This can be seen as a uniform projection of the relevance $R_j^{(l+1)}$ onto the receptive field of neuron j. To the more bottom layers we apply the $\flat$-decomposition rule, the less *deep* the relevance propagation becomes (as seen from the network output). Relevance explanations then correspond to concepts of a higher semantic levels, as represented by the model's neurons in those inner layers: Figure 5.2 (right) demonstrates the relationship of relevance decomposition depth to relevance map semantics and local scale at hand of a GoogleNet model trained to predict gender. When performing LRP according to Section 5.4.1 through all layers of the DNN (high decomposition depth), relevance maps identify local structures influencing the model in favour or against an output class of choice, such as the *shape* of the nose and mouth. The further up the network we choose the layer from which on to uniformly distribute the relevance of a neuron j to all its inputs i according to Equations (2.27) and (5.1) (*i.e.* the lower the decomposition depth), the less fine grained the relevance attribution becomes. LRP begins to identify *regions of compound features*, such as the area around chin and mouth or the wig as discriminating features, with increasing receptive field sizes of hidden neurons.

Note though, that Equations (2.27) and (5.1) might not be a meaningful decomposion to all types of neural network layers. While component-wise activation layers for example are not affected, an application to fully connected layers will yield – due to the layer's all-to-all connectivity – completely uniform and thus non-sensical relevance maps. An alternative decomposition suitable for fully connected layers is the w^2-rule from (Montavon et al., 2017).

Firstly, we can use this decomposition rule to control the structural and semantic scale of image characteristics which should be considered for relevance attribution (onto pixel level, per model) and secondly, we can make use of it to ensure the choice of a common scale for the facial attributes considered during the explanation of the model decision for all DNNs. Lastly, the decomposition rule in Equations (2.27) and (5.1) introduces an invariance to the normalization of the given input samples. Thus, we apply the $\flat$-rule to the first convolution layer ("`conv1`") of the CaffeNet model. For the GoogleNet model, the rule is applied to all convolution layers closest to the input and below the lowest inception block (*i.e.* all layers up to and including layer "`conv2/3x3`"). For the VGG-16, all layers up to (and including) the second pooling layer ("`pool2`") are decomposed using Equation (5.1).

Note that for Bag of Words models in Section 2.4.2, the $\flat$-decomposition rule has been applied to the decomposition of local feature relevance scores $R_l^{(2)}$ to pixel-level relevance scores $R_p^{(1)}$ out of necessity, since in general there is no clear (weighted) relationship between individual pixels and local descriptor dimensions, *e.g.* as it is the case with the quantile-based descriptors used in (Binder et al., 2013). For some Bag of Words model pipelines such as the Fisher vector SVM with SIFT descriptors from Chapter 4 it is possible to further apply LRP wrt to the computed local descriptor computations. Details and examples are given in Appendix D.

5.5 Model Evaluation and Results

We score all trained models using the oversampling evaluation scheme (Levi et al., 2015), by using the average prediction from ten crops (four corner and one center crop, plus mirrored versions) per sample. Results for age and gender prediction are shown in Table 5.2. The columns of both tables correspond to the described models; the **A**dienceNet, **C**affeNet, **G**ooglenet and **V**GG-16. Following previous work we also report 1-off accuracy results – the

TABLE 5.2: **Face categorization results in accuracy and percent, using oversampling for prediction.** *Left:* Results for **age** classification. Small numbers next to the accuracy score show 1-off accuracy, *i.e.* the accuracy of predicting the correct age group *or* an adjacent one. *Right:* Results for **gender** prediction. Bold values match or exceed the currently reported state of the art results from (Dehghan et al., 2017) on the Adience benchmark dataset.

age	A	C	G	V
[i,·]	51.4 87.0	52.5 87.9	54.4 89.3	–
[r,·]	51.9 87.4	52.6 89.0	54.4 90.0	–
[m,·]	53.6 88.4	54.4 89.7	56.5 90.8	–
[i,n]	–	51.7 87.6	56.6 91.0	53.8 88.2
[r,n]	–	52.2 87.1	57.5 92.0	–
[m,n]	–	53.0 88.4	58.8 92.7	56.5 90.0
[i,w]	–	–	–	60.2 94.2
[r,w]	–	–	–	–
[m,w]	–	–	–	63.0 96.0

gender	A	C	G	V
[i,·]	88.1	87.7	88.2	–
[r,·]	88.3	88.0	89.3	–
[m,·]	89.0	88.9	89.7	–
[i,n]	–	90.0	**91.2**	**92.0**
[r,n]	–	90.7	**91.7**	–
[m,n]	–	90.6	**92.0**	**92.7**
[i,w]	–	–	–	90.6
[r,w]	–	–	–	–
[m,w]	–	–	–	**92.3**

accuracy obtained when predicting at least the age label adjacent to the correct one – for the age prediction task.

The row headers describe the training and evaluation setting: A first value of [i] signifies the use of [i]n-plane face alignment from (Eidinger et al., 2014) as a preprocessing step for training and testing, [r] stands for [r]otation based alignment and [m] describes results obtained when both rotation aligned and in-plane aligned images have been [m]ixed for training and images from the [r] test set have been used for evaluation. Second values [n] or [w] describe weight initialization using Image[n]et and IMDB-[w]IKI respectively. No second value (shown as a dot: [·]) means the model has been trained from scratch with random weight initialization.

The results in above tables list the measured performance after a fixed amount of training steps. Intermediate models which might have shown slightly better performance are ignored in favour of comparability. With our attempt to replicate the results from (Levi et al., 2015) based on the code provided by the authors, we managed to exceed the reported results in both accuracy by (+1.2%) and 1-off accuracy (+2.7%) for age prediction and accuracy (+1.5%) for gender prediction. As expected, the structurally comparable CaffeNet architecture obtains relatable results for both learning problems with random model weight initialization. We then further compared the relatively fast to train CaffeNet model to the GoogleNet model in all data preprocessing configurations when trained from scratch and fine-tuned based on the ImageNet weights. We try to replicate the measurements from (Rothe et al., 2016) to verify the observations made based on the other models. Here, we did not fully manage to reach the reported results, despite using the model pre-trained on the IMDB-WIKI data as provided by the authors. However, we closely scrape by the reported results with slight differences in both accuracy (−1.0%) and 1-off accuracy (−0.6%), averaged over all five splits of the data with a model trained on the mixed training set. In all evaluated settings shown in Figure 5.3 we can observe overall trends in choices for architecture, dataset composition and preprocessing and model initialization.

5.5.1 Remarks on Model Architecture

In all settings, the CaffeNet architecture is outperformed by the more complex and deep GoogleNet and VGG-16 models. For gender classification under comparable settings, the best VGG-16 models outperform the best GoogleNet models. Figure 5.4 visualizes based on which characteristics of the given input samples the classifiers predict gender.

We observe that model performance correlates with network depth, which in turn correlates with the structure observable in the heatmaps computed with LRP. This result is in accord with our observations and measurements on th CIFAR-10 dataset in Section 3.5.4 and on Pascal VOC and ImageNet data in Section 4.5. Here, for instance, all models recognize

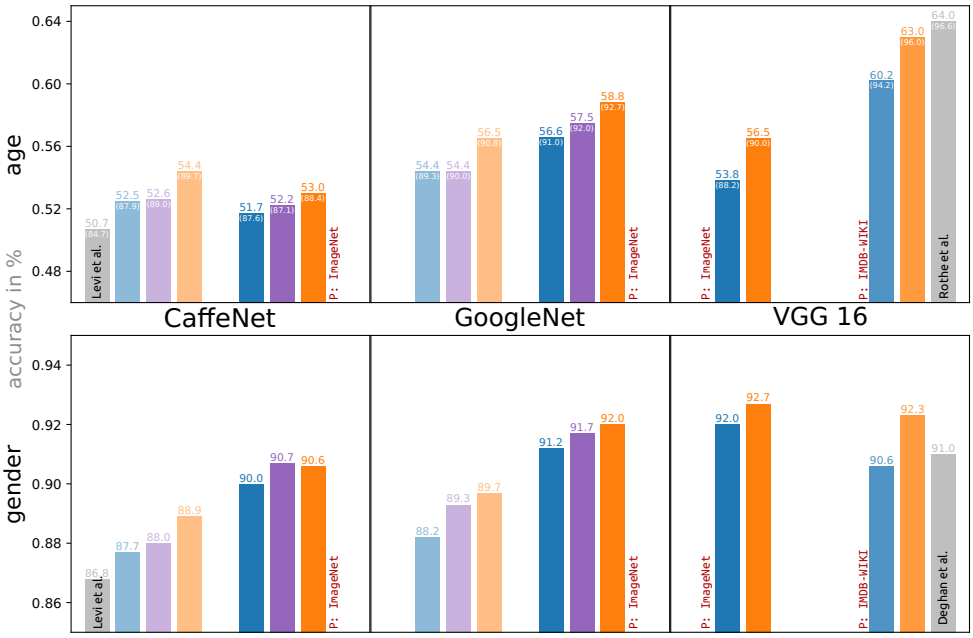

FIGURE 5.3: **Graphical comparison of model results for age and gender prediction.** *Top to bottom:* Age prediction and gender prediction as accuracies in percent. *Left to right:* Results for the CaffeNet, GoogleNet and VGG 16 DNNs. The gray bars to the left and right show the respective earliest and best results as reported in (Levi et al., 2015; Rothe et al., 2016; Dehghan et al., 2017) for each prediction problem as reference. Color coding corresponds to data preprocessing and shading to model initialization: Red text next to groups of opaque bars: Method of pretraining. IMDB-WIKI pretraining is preceded by ImageNet pretraining. Transparent bars show result for randomly initialized models. Blue bars: Affine [i]n-plane alignment. Violet bars: [r]otation alignment. Orange bars: [m]ixed training data. All results are averaged over the five splits of the Adience data set.

female faces dominantly via hair line and eyes, and males based on the bottom half of the face and uncovered ears. The CaffeNet model tends to concentrate more on isolated aspects of a given input compared to the other two, especially for men, while being less certain in its prediction, reflected by the more pronounced concentrations of negative relevance, indicating strong influence of contradicting input features to the model output analyzed via LRP.

5.5.2 Observations on Preprocessing

For all three models, we can observe the overall trend for both prediction problems, that the in-plane alignment preprocessing step is not beneficial to classifier performance, compared to rotation alignment. We reason the better performance on only rotation aligned images to be justified in the potential of and for DNNs to learn for the domain of face images canonically meaningful sets of features. For the face images aligned using the technique presented in (Eidinger et al., 2014), this is more difficult. Especially for images of children, the faces aligned to reference frames suitable for adults result in head shapes of uncharacteristic aspect ratios for the age group or even faulty alignments. Section D.2 in Appendix D presents additional results regarding the effect of facial alignment preprocessing to the age recognition task for all three models and draws conclusions regarding a combined effect with dataset composition to the model performance per class. Figure 5.5 demonstrates the nature of this artefactual noise introduced to the data by unsuitable or incorrect alignment.

All models benefit the most from combining both the rotation aligned and the landmark aligned datasets for training. For one, this effectively increases the size of the training set twofold, but also – perhaps more importantly – allows the learning of a more robust feature

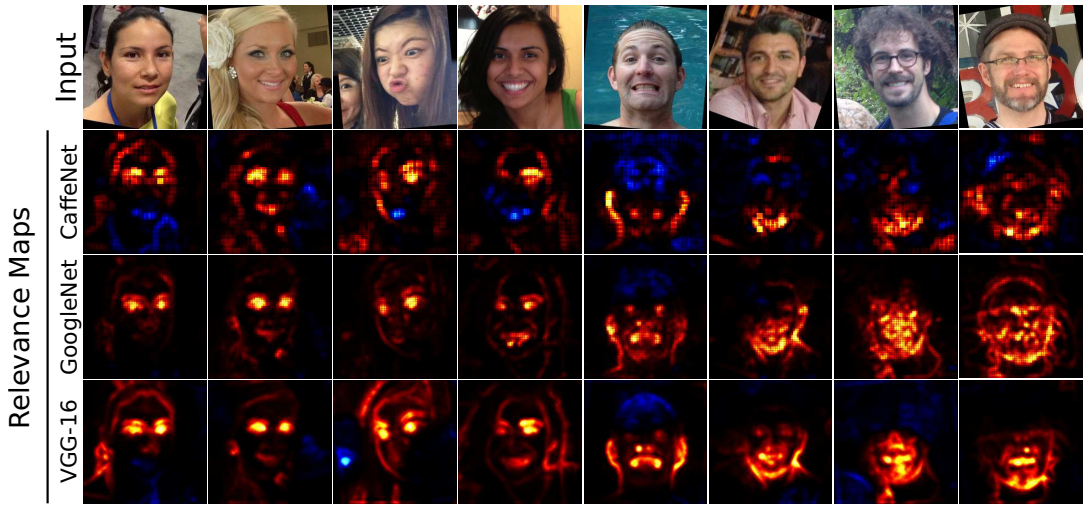

FIGURE 5.4: **Different gender prediction strategies of different models.** *Top:* The input samples provided to the models. *2nd* row to last row: Relevance maps for the CaffeNet, GoogleNet and VGG-16 networks. All models have been pre-trained on ImageNet and predict gender correctly for the given samples. Visually smoother (*i.e.* less blocky) heatmaps are a consequence of smaller filters and stride in the bottom layers and thus smoother local averaging of relevances across multiple layers via the $\flat$-decomposition rule.

TABLE 5.3: **Test set swapping results.** *Left:* Results for evaluated **age** classifiers. *Right:* Results for evaluated **gender** classifiers. Performance is considerably worse when the *incorrect* preprocessing is used for testing, due to overfit feature sets: Models trained on [i]n-plane aligned samples have been tested on rotated test samples. Models trained on [r]otation aligned training sets have been evaluated on samples affected by [i]n-plane alignment. Pretraining (here, on Image[n]et) can yield robust model parameters, compensating for the deviating test statistics.

age	**A**	**C**	**G**
[i]	40.8 $_{75.4}$	40.3 $_{76.3}$	44.6 $_{80.8}$
[r]	46.9 $_{82.8}$	46.1 $_{82.5}$	46.4 $_{83.2}$
[i,n]	–	45.2 $_{82.02}$	49.4 $_{87.2}$
[r,n]	–	48.8 $_{84.9}$	53.6 $_{89.9}$

gender	**A**	**C**	**G**
[i]	81.1	80.5	83.5
[r]	81.3	84.6	86.0
[i,n]	–	84.5	89.6
[r,n]	–	88.5	90.0

set: The models trained on a combination of both the landmark aligned and rotation aligned images perform well on test sets resulting from both preprocessing techniques. The results for models trained on the combined training sets in Table 5.2 (rows marked with [m]) show performance measurements evaluated on the rotation aligned test set only. Performance measurements on in-plane aligned data are with a difference of $< 1\%$ only insignificantly lower and thus omitted from the table.

In order to underline the effect of increased robustness of the models trained on the more diverse [r]oration aligned training set we evaluated models trained on [i]n-plane aligned images with [r]otation aligned test images and vice versa. Corresponding model performances are listed in Table 5.3. Some models trained on data prepared with one alignment technique evaluated against the test set of the other perform even worse than the early SVM-based models from (Eidinger et al., 2014), despite their competitive results from the combined training set. The models trained on the in-plane aligned images have more difficulty predicting on the unseen setting than the models trained on the only rotated images, where the original facial pose and the proportions of the face image are mostly preserved.

For the VGG-16 model, we compared the in-plane alignment to the mixed training set – the worst to the best expected results. Here again, the mixed training data results in a better

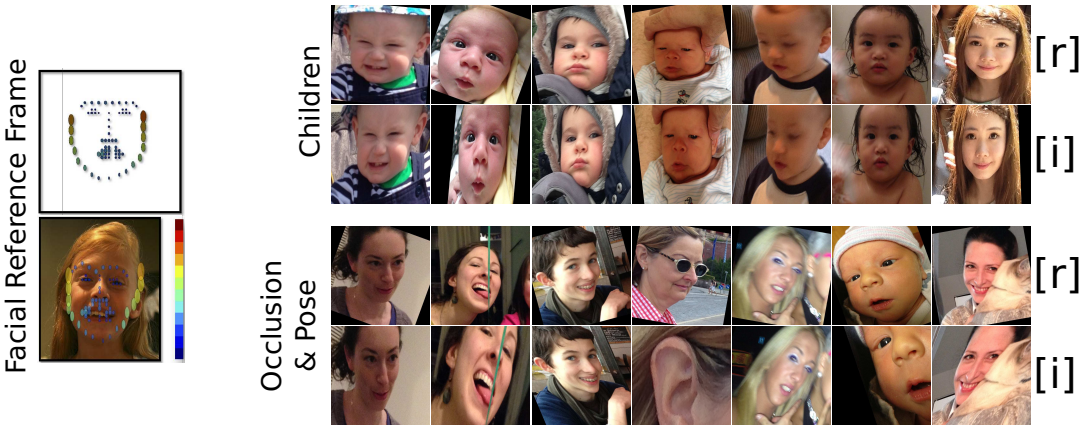

FIGURE 5.5: **Failure cases from the in-plane aligned version of the Adience dataset.** *Left:* Schematic view of the reference frame for facial landmarks (Image source: (Eidinger et al., 2014)) and its application. *Top right:* Failure cases of the alignment process with face images of children. *Bottom right:* Failure cases of the alignment process due to pose or occlusion. In both groups of images the top row shows [r]otation aligned images and the bottom row [i]n-plane aligned versions. The landmark based alignment technique attempts to morph the image such that the shown face aligns with a reference frame of facial landmarks fitting adult facial proportions. The left six face images showing children are taken from the age group of (0-2) which are classified correctly under rotation alignment and are placed at least one age group above by the predictor under landmark-based alignment. The in-plane alignment technique applied to one variant of the Adience dataset tends to elongate faces vertically.

model than when only in-plane alignment is used. Figure 5.3 provides an overview over all results.

5.5.3 Observations on Initialization

We find that the GoogleNet model responds well to fine-tuning on the weights pre-trained on ImageNet and responds with an increase in performance for both classification problems and in all dataset configurations. The CaffeNet, however, slightly loses performance when fine tuned for age group prediction, while benefiting in gender prediction. The better response of the GoogleNet compared to the CaffeNet, when initialized with their respective ImageNet weights might be caused by the quality of the initial parameters: While the GoogleNet achieves a 6.7% top-5 error on ImageNet, the CaffeNet only reaches 16.4%. Evaluating on the *incorrectly preprocessed* test data (Table 5.3), both fine tuned models trained on rotation aligned images manage to recover their respective performance ratings compared to models trained from scratch and being evaluated on the *correctly preprosessed* data. The GoogleNet model even exceeds the performance of the same architecture initialized randomly but both trained and evaluated on the rotated images.

The measurable beneficial effect of appropriate pretraining reflects in the relevance maps in Figures 5.6 and 5.7: For the gender recognition task with the GoogleNet model, ImageNet pretraining leads to the use of larger and meaningful parts of the face for prediction, while the randomly initialized model picks out single characteristics during training which correlate the most with the target class. This includes eyebrows and lips defining female faces and nose, chin and uncovered ears for men. This leads *e.g.* to thinner eyebrows in male faces being regarded as contradictory evidence for class “male”.

We see comparable results for the VGG-16 and GoogleNet models on age group estimation in Figure 5.7 when comparing pretraining on ImageNet, IMDB-WIKI and random initialisation. Again, the randomly initialized GoogleNet model picks out isolated facial features for age prediction, while its ImageNet-pretrained counterpart is able to use multiple face regions in combination for prediction. The model initialized with IMDB-WIKI weights,

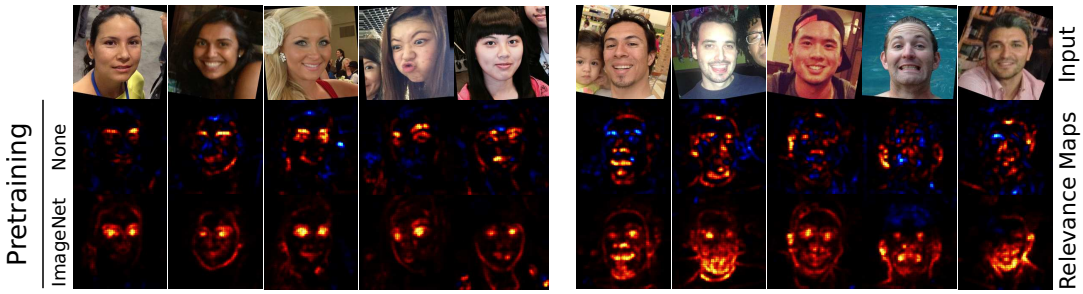

FIGURE 5.6: **The effect of pretraining for gender recognition on GoogleNet visualized via relevance maps.** The finetuned model predicts based on an ensemble of facial features, whereas the model starting with random weights has overfit on an isolated set of features characteristic to the target classes.

with the pretraining task being age estimation on 101 age categories, concentrates more on the facial features themselves, while the ImageNet-initialized one is more prone to distraction from background elements and clothing items.

For the problem of gender recognition, the VGG-16 is affected less from weight initialization than from the quality of data preprocessing. Here, IMDB-WIKI pretraining might have an only diminished effect due to firstly the ImageNet weights providing an already good set of starting weights and secondly, the pretraining objective (age recognition) being orthogonal to the task of gender recognition. In fact, other than for age recognition, the VGG-16 models initialized with ImageNet weights converged to better parameters than their counterparts.

5.6 Limitations

Our analyses via relevance maps are limited to the *localization* of relevant facial features in the input images. A comparison of both preprocessing (alignment) choices via relevance maps is thus more difficult and did not yield any clear insights, due to the at times extreme differences in the processed face images (see Figure 5.5).

In Section 5.4.1 we have proposed improvements to rule parameterizations for sequential DNN models, which was sufficient for this chapter and the models from referenced literature. With the decomposition rules described in Section 2.3, however, LRP can already be applied to Neural Networks with more densely interconnected and non-sequential layer structure, such as DenseNet (G. Huang et al., 2017) or ResNet architectures (He et al., 2016; Zagoruyko et al., 2016). Further investigations are required in order to assess the applicability of LRP to these model architectures.

The values for parameters α, β and ϵ used throughout this chapter have been determined empirically. While yielding good results qualitatively and quantitatively (Samek et al., 2017; Kohlbrenner, 2017; Chapter 3), future research should be dedicated to the analytical inference of optimal parameters for decomposition, *i.e.* wrt criteria which are yet to be determined.

5.7 Conclusion

Recent Deep Neural Network models are able to accurately analyze human face images, in particular recognize the persons' age, gender and emotional state. Due to their complex non-linear structure, however, these models often operate as black-boxes and until very recently it was unclear *why* they arrived at their predictions. In this chapter we opened the black-box classifier using Layer-wise Relevance Propagation and investigated which facial features are actually used for age and gender prediction under different conditions: We compared different image preprocessing, model initialization and architecture choices on the challenging Adience dataset and discussed how they affect performance. In order to compare model

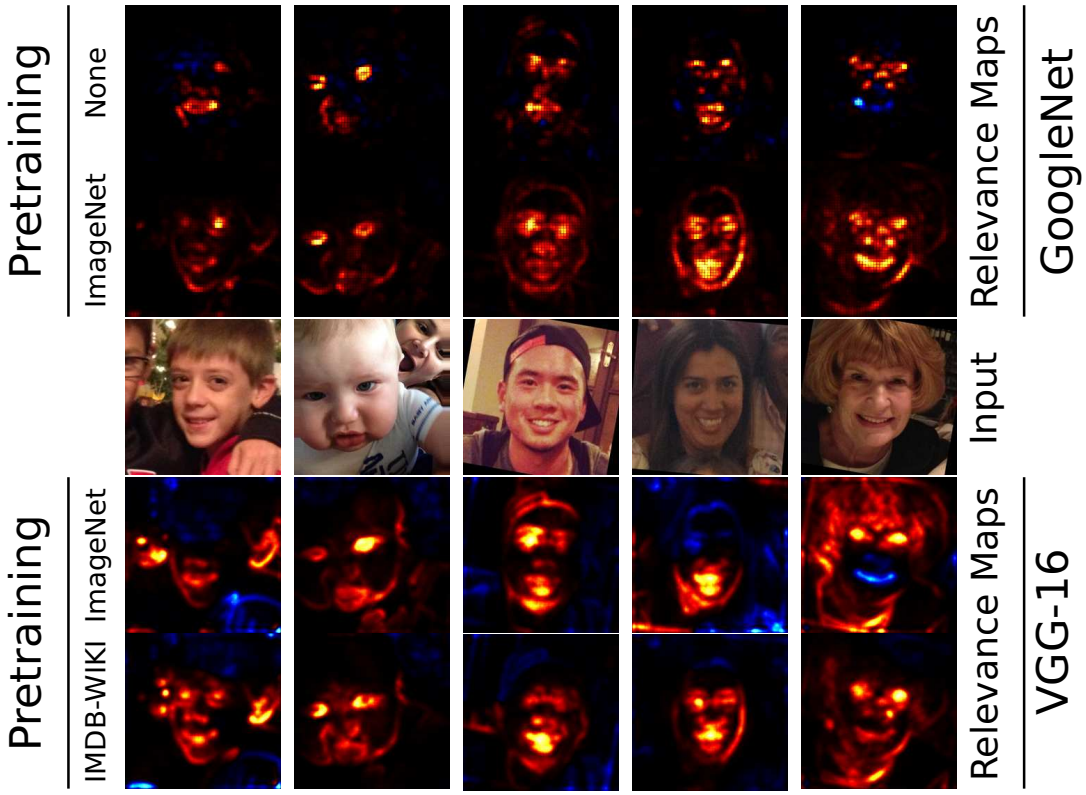

FIGURE 5.7: **The effect of pretraining for age recognition on GoogleNet and VGG-16 visualized via relevance maps.** The randomly initialized GoogleNet model uses isolated facial features for prediction. Both models pretrained on ImageNet use larger regions and ensembles of face characteristics. The VGG-16 model pre-trained on IMDB-WIKI, *i.e.* images of faces, uses facial information exclusively, while its ImageNet-pretrained counterpart tends to use auxiliary information such as clothing items for age prediction.

decisions via relevance maps between different DNN models at the same scale, we introduce the $\flat$-decomposition rule and propose a setting of LRP decomposition rules suited for sequential DNN architectures.

By using LRP to visualize the models' interactions with the given input samples, we demonstrate that appropriate model initialization via pretraining counteracts overfitting, leading to a holistic perception of the input. We also demonstrate that appropriate model initialization via pretrained weights can be used to improve the robustness of the trained predictor against distorting noise and avoids overfitting on only a few isolated input characteristics which reflects both in prediction performance and explanatory relevance maps. With a combination of simple preprocessing steps, we achieve state of the art performance for gender classification on the Adience benchmark data set.

Chapter 6

Ensemble Relevance Map Analysis of Classifier Behaviour

In previous chapters, relevance maps have been used to analyze predictions made for individual samples over a range of model types and architectures. While LRP offers a solution to gain insight into the decision process of a model while maintaining full model expressiveness for the prediction task, a manual analysis of hundreds or thousands of relevance maps can prove to be too laborous and time consuming as a feasible means for understanding the strategies learned by a model exhaustively. In this chapter, we introduce a novel approach based on spectral cluster analysis for systematically and automatically analyzing the insights gained via individual relevance maps over large sets of data. The contents of the following sections are previously unpublished technical constributions and analyses, and will be subject to an upcoming publication (Lapuschkin et al., 2018).

6.1 Introduction

This chapter demonstrates the usefulness of relevance maps as features for analyzing classifier behaviour automatically. The proposed approach constitutes a new tool for semi-automatic model analysis – based on the computation of relevance maps and spectral analysis – which closes the gap between performance evaluations over whole datasets (no human inspection needed, but no information about the inner working of the classifier) and visual assessment of single predictions (information about the inner working, but requires human inspection thus cannot be performed for whole datasets). We name this method *Spectral Relevance Analysis* or short; SpRAy. Technically, SpRAy allows to detect predictions based on irregularly frequent reoccurrence of non-obvious and highly similar image features, which can assist in the uncovering of intriguing or suspicious patterns in either the data used for training (and testing), the prediction strategies learned by the model, or both. The identified relevant features may be benign and truly meaningful representatives of the object class of interest, or they may be co-occurring features learned by the model but not intended to be part of the class, and ultimately of the model's decision process. In the latter case, SpRAy will assist researchers and software engineers to systematically find weak points in their trained models or training datasets, or even generate new knowledge about the data and problem solving strategies.

In the following, we
(1) introduce the methodology behind SpRAy, based on Spectral Analysis of Relevance maps in Section 6.2.
(2) The method is then applied to the Fisher Vector and DNN models from Chapter 4 in Section 6.3, where it is able to re-identify the prediction artefacts already found by manually inspecting numerous relevance maps and systematically reveals previously unknown and interesting model (mis)behaviour of the DNN predictor.
(3) Via these newly gained insights we are able to form and verify a hypothesis regarding the cause of the previously unidentified prediction artefact in Section 6.3.1, providing an opening for iterative improvement of the analyzed model. Section 6.5 concludes the chapter with an outlook towards future development of the method.

6.2 Spectral Relevance Analysis

The predictions made by the FV model in Chapter 4 for class "horse" are based on image features frequently appearing in a very structured manner in the object class' image data, despite not describing the animal in any way visually. The artifact in question is a copyright tag in approximately one fifth of the test and training images representing the object class, which has been picked up by the model as an even stronger indicator of "horseness" than the pictured horses themselves. Since the copyright watermark is already present in the input data, a *very* attentive human observer might have noticed it, yet since those samples found their way into the Pascal VOC (Everingham et al., 2009) data, which also was widely used as a benchmark dataset for several years, we suppose ultimately no one did. Humans often solve tasks in certain ways predisposed by life (long) experience. Any assessors of the data might thus have directed their attention towards the intended objects (known through life experience) shown on the image samples instead of auxiliary and unexpected features. Machine learning algorithms on the other hand can only connect information which is available in the discrete samples of finite example datasets.

With the goal to systematically identify pitfalls in the data a model might – or in fact does – fall victim to, we establish as *Spectral Relevance Analysis* (SpRAy) an analysis pipeline under the assumption that certain secondary structures and features are in some way (*e.g.* feature representation) correlated with a target class, rendering them dominant identifiers for that class to a trained model, although those features might not be obvious to a human observer at all, or appear during corellation analysis between pixel space and the target label. If said features are very descriptive of a prediction target or object class, their presence is expected to result in concentrated relevance feedback within the corresponding models' and samples' relevance maps.

For our analyses, we employ a sequence of three steps: (1) (relevance map) input (size) preprocessing. (2) Spectral cluster analysis (SC) and (3) complementarily; t-Stochastic Neighborhood Embedding (t-SNE) for visualizating the results. Figure 6.1 visualizes the workflow of SpRAy.

The initial preprocessing step (1) renders all relevance maps uniform in shape and size, creating viable inputs for the following application of SC and t-SNE. Some predictors such as the FV classifier can accept as inputs images of arbitrary size, making this step a necessity on some cases. Also, by downsizing the input relevance maps, *e.g.* by pooling relevance spatially wrt a regular grid, the dimensionality of the analysis problem is decreased (In general, this step is practically usefull but not absolutely necessary). This expedites the process considerably and produces more robust results. Suitable preprocessing choices can be used to focus the following steps on certain characteristics within the input relevance maps.

The SC (Meila et al., 2001; Ng et al., 2002; Von Luxburg, 2007) used in step (2) is a clustering algorithm with interesting analytic properties, which constitutes the foundation of the analysis pipeline. The method relies on the computation of a weighted *affinity* or *adjacency* matrix $W = (w_{ij})_{i,j=1,\dots,N}$, measuring the similarity $w_{ij} \geqslant 0$ between all N samples s_i and s_j of a dataset. *E.g.* when building affinity matrices based on k-nearest-neighborhood relationships as we do in Section 6.3, we compute w_{ij} as

$$w_{ij} = \begin{cases} 1 & \text{if } s_j \text{ is among the k nearest neighbors of } s_i \\ 0 & \text{else} \end{cases} \tag{6.1}$$

Since this is an asymmetrical relationship between s_i and s_j we follow (Von Luxburg, 2007) and create a symmetric affinity matrix W with $w_{ij} = \max(w_{ij}, w_{ji})$, describing an undirected adjacency graph between all samples. The evaluations in Section 6.3 (which are based on binary neighbourhood relationships) yielded highly similar results when using (euclidean) distances $\|s_i - s_j\|$ between connected neighbours instead of above binary neighbourhood relationships, with only small differences in eigenvalue spectra.

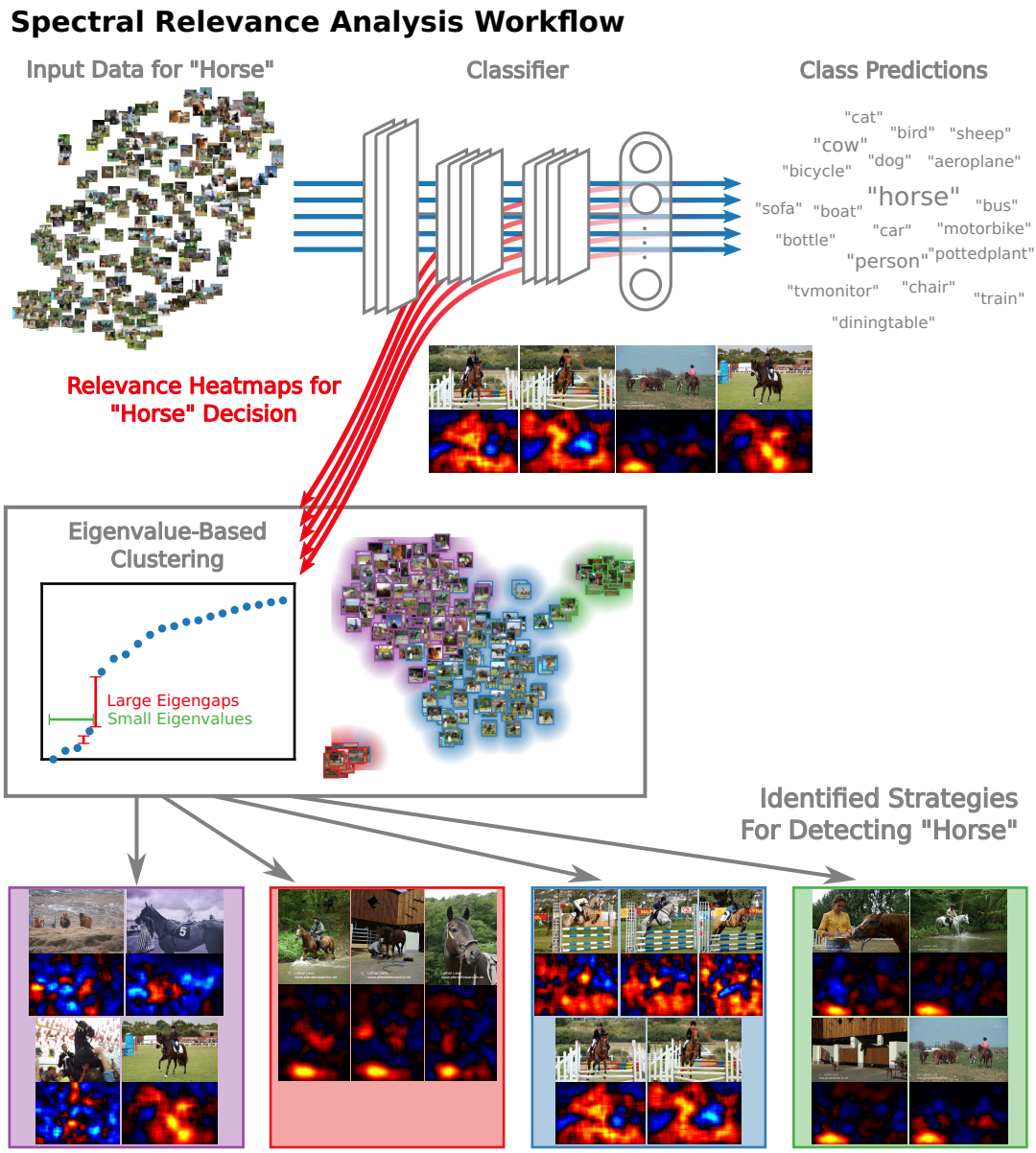

FIGURE 6.1: **The workflow of Spectral Relevance Analysis.** First, relevance maps are computed for data samples and object classes of interest, which requires a forward and a LRP backward pass through the model. Then, an eigenvalue-based spectral cluster analysis is performed to identify different prediction strategies within the analyzed data. Visualizations of the clustered relevance maps and cluster groupings supported by t-SNE inform about the nature of the prediction strategies, and whether they are benign or a bias in the model's prediction, requiring human intervention in order to obtain a well-generalizing model.

From the matrix W, a graph Laplacian L is then computed as

$$\begin{aligned} d_i &= \sum_j w_{ij} \\ D &= \mathrm{diag}([d_1, \ldots, d_N]) \\ L &= D - W \end{aligned} \tag{6.2}$$

with D being a diagonal matrix with entries d_{ii} describing the *degree* (of connectivity) of a

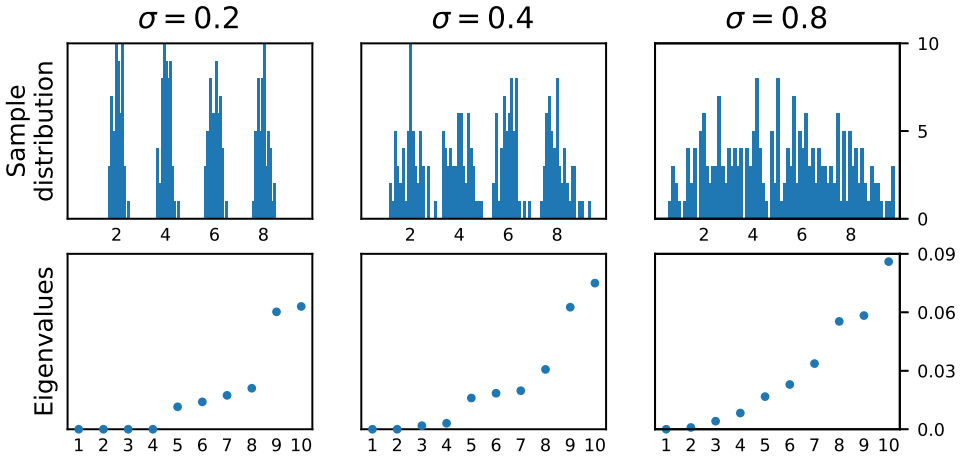

FIGURE 6.2: **Histograms of datasets containing samples from standard normal distributions with corresponding standard deviations σ, centered with $\mu \in \{2, 4, 6, 8\}$ and the 10 smallest eigenvalues each.** Inspired by Figure 4 from (Von Luxburg, 2007).

sample i. Performing an eigenvalue decomposition on the Laplacian L then yields N eigenvalues $\lambda_i, \ldots, \lambda_N$, with the guarantee of $\min_i \lambda_i = 0$, and a set of corresponding eigenvectors. Here, the number of eigenvalues $\lambda_i = 0$ identifies the number of (completely) disjoint clusters within the analyzed set of data. For non-empty datasets, there is at least one cluster and thus the guarantee for at least one eigenvalue to be zero-valued. The final step of cluster label assignment can then be performed using an (arbitrary) clustering method – *e.g.* k-means – on the N eigenvectors. On non-synthetic data, however, there seldomly are cleanly disjoint groups of samples, and the (main) cluster densities usually are at least weakly connected. In that case, the main densities can be identified by eigenvalues *close to zero* as opposed to *exactly zero*, followed by an *eigengap*. The eigengap is a sudden increase in the difference between two eigenvalues in sequence; $|\lambda_{i+1} - \lambda_i|$. In Figure 6.2 we recreate an example from (Von Luxburg, 2007) which demonstrates the change in the eigenvalue spectrum with increasing overlap between the sampled distributions.

With $\sigma = 0.2$, four non-overlapping sample sets have been generated. The four smallest eigenvalues of this dataset thus are exactly zero and the eigengap is distinctive. With increasing overlap of the sampled distributions ($\sigma = 0.4$), the eigengap still indicates four clusters of data clearly, although the four smallest eigenvalues indicating well defined groups of samples are not exactly equal to zero anymore. Once the distributions overlap considerably ($\sigma = 0.8$), there is no well-distinguished eigengap anymore to identify the number of densities sampled from truthfully. We can use the eigengap property of an analyzed dataset to identify structural similarities in the image input data of an object class, but also the corresponding relevance maps computed for each sample of the dataset wrt to a trained model, reflecting the model's prediction strategy.

In step (3) t-SNE (Maaten et al., 2008) is used to compute a two-dimensional (or in general, a human-interpretable lower dimensional) embedding of the analyzed data, based on pair-wise distances between samples. The resulting embedding is used to visualize groups of input images or relevance maps by similarity, aiding in the identification of interesting patterns in the models' prediction. Properties of the t-SNE are known to be connected to cluster assignments computed via SC (Linderman et al., 2017). Sample to sample relationships computed during the SC step can be re-used in the t-SNE embedding phase by transforming the affinity matrix W to a distance matrix, *e.g.* as $\text{DIST} = \frac{1}{W+\epsilon}$ for above binary k-nearest-neighbor affinity matrix by adding a small ϵ to the denominator to encode high distances between (via k-nearest-neighbors) unconnected points.

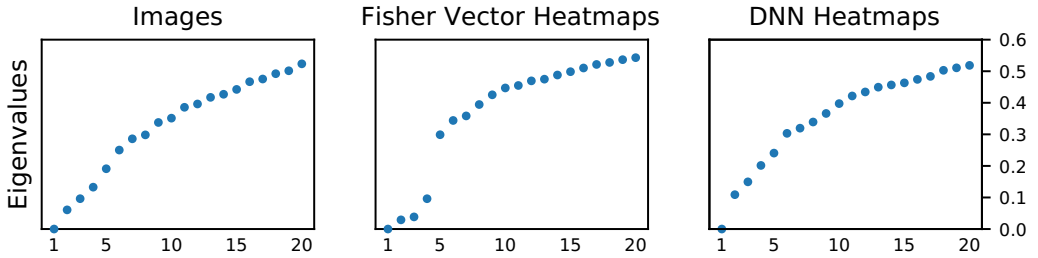

FIGURE 6.3: **The first 20 eigenvalues for class "horse" for images, relevance maps for the Fisher Vector predictor and the DNN respectively.**

6.3 Uncovering Prediction Strategies of the Pascal VOC Classifiers

Searching to uncover distinct prediction strategies within the scope of single object classes, we perform analyses for all classes of the Pascal VOC 2007 test set separately. Given the extensive amount of options to configure the SC algorithm, we perform our analysis using the recommended parameters from (Von Luxburg, 2007), *i.e.* we build neighborhood graphs using the k-nearest neighbor algorithm with $k = \log(n)$ and n being the number of samples and use normalized, symmetric and positive semi-definite Laplacians (Ng et al., 2002) L_{sym} instead of the unnormalized L. The use of L_{sym} also allows for meaningful comparisons between input sample sets of different sizes, which will be important in a moment with Figure 6.5. For t-SNE, we set the perplexity to 7, the parameter for early exaggeration to 6. The method is known to be insensitive to small changes in both input parameters. For downsizing images, we use standard interpolation algorithms for images and sum pooling over a regular grid of pixels for relevance maps to reduces the inputs to (20×20) sized matrices, representing local relevance allocation behaviour throughout the dataset. Our experiments have shown that the qualitative results as presented in below setting can also be reliably obtained for input sizes (5×5), (10×10) and (50×50).

In the following we inspect the eigenvalue spectra for all classes and both the FV and DNN model from Chapter 4. The relevance maps for the Fisher vector model have been computed in the same manner as in Chapter 4. Relevance maps for the DNN model are based on the improved approach for sequential network models[1] introduced in Chapter 5 and (Kohlbrenner, 2017). Figure 6.3 shows the eigenvalue spectrum for class "horse" for input images and relevance maps for both the FV and DNN model. Previous manual inspection has revealed that the FV predictor primarily relies on a copyright watermark embedded into the pixels of some images for predicting the class. The gaps after λ_3 and λ_4 in the eigenvalue spectrum of the FV model's relevance maps are apparent and indicate multiple well-separated clusters. In the eigenvalue spectra of the images and the relevance maps obtained with the DNN model, the largest gap between eigenvalues occurs after λ_1, indicating only one densely connected cluster over all test samples for that object category.

Likewise to the eigenvalue spectra in Figure 6.3, the cluster label assignments computed via SC as displayed in Figure 6.4 clearly show point clouds of homogeneous density for both DNN relevance maps and input images and well separated clusters for the FV relevance maps. Figure 6.3 (bottom) also demonstrates that the clusters found with SC (color coded) in the input data correspond well to the embeddings computed by t-SNE (embedding locations in $\mathbb{R}^2$) from the same input data with very high similarity. The connection between SC and t-SNE described in (Linderman et al., 2017) can also be observed in all following results, making t-SNE a suitable tool for visually supporting the analysis via SC.

[1] $\alpha\beta$-LRP with $\alpha = 2, \beta = -1$ is applied to convolution layers, and ϵ-LRP to with $\epsilon = 0.01$ to dense layers. For the model's input layer we use the $\flat$-decomposition rule. These settings make the decomposition with LRP invariant to normalizations in input space and remove the assumption that $f(x) \geqslant 0$ always due to the $\alpha\beta$-rule applied to the dense output layers (Montavon et al., 2016; Montavon et al., 2017).

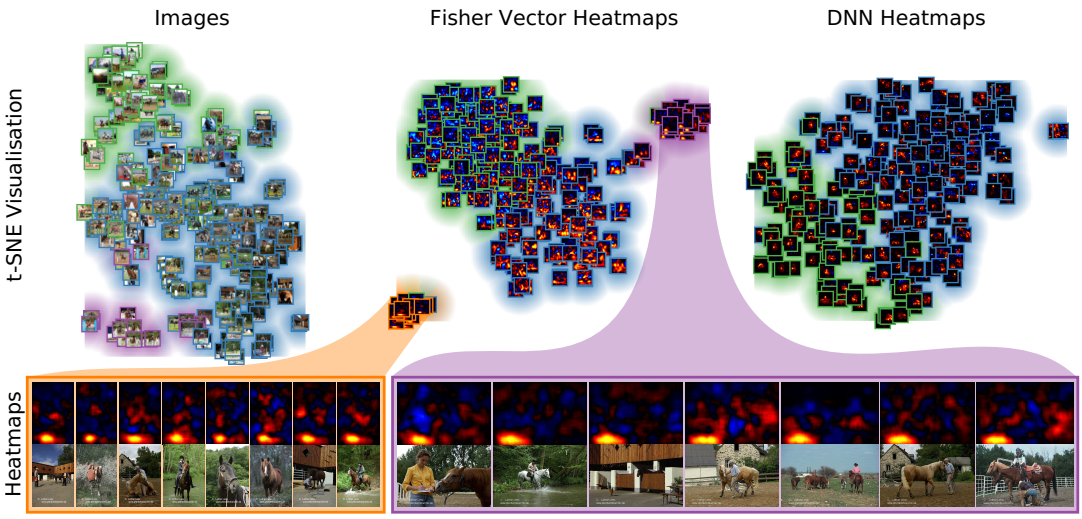

FIGURE 6.4: **Cluster label assignments for class "horse" via SC computed from input images (left), FV relevance maps (middle) and DNN relevance maps (right).** Embedding coordinates in $\mathbb{R}^2$ for visualization have been computed on pair-wise distances derived from the weighted affinity matrix W used for SC. *Top row:* The data (images or relevance maps) used as input for the analysis at their respective two dimensional embedding coordinates. The coloured borders and aurae around the input samples show cluster label assignments via SC. *Bottom row:* The enlarged images and FV heatmaps (without preprocessing) show examples grouped into the bottom left cluster (portrait oriented images) and the top right cluster (landscape oriented images), revealing the FV models' strong reaction to the copyright watermark.

While for the DNN relevance maps the embedding and clustering results seem to be grouped by dominant combinations of horse and rider in the foreground vs. the presence of contradictory image features, the results for color images as input features are determined by visual scene similarity in structure and color composition (Figure 6.4 (top)). For the Fisher Vector relevance maps, the presence of two well defined groups besides the main cluster can be identified by both the cluster labels assigned via SC and also the embedding locations computed with t-SNE. This result suggests that the FV model picks up on information which is apparently correlated strongly with the target label "horse", but not obvious from the input images alone. Both clusters separated from the main point cloud contain relevance maps for image samples with the copyright watermark in question, which only covers a small number of pixels within the images originating from the same horse-themed online image repository.

With the application of SpRAy on the input images and relevance maps computed for both models, we have gained a clue that the FV model picks up on input information in a structured and consistent way, which is not obviously present in the image domain and not used by the DNN model. That is, on relevance maps we obtain clusters of samples which are predicted with a very similar localized strategy by the FV model, which we do not already find among the input samples. The model has learned a prediction strategy suitable for a small subset of training and test samples based on an image feature hard to find on the images directly.

To avoid further manual analysis of each image and its corresponding relevance map we proceed to close the gap between the assessment of individual predictions (via relevance maps) and performance measures over whole datasets by conducting above spectral and embedding analysis for all classes of the Pascal VOC dataset. We then compare the spectrum of the first $k = 5$ eigenvalues for input images, FV relevances and DNN relevances in search for classes which are candidates for suspicious samples or learned prediction strategies. Figure 6.5 gives an overview over the measured eigenvalue spectra.

We can observe, that the class "horse" appears first in the ranking of FV based eigenvalues, at fourth position for images and position six for DNN based eigenvalues. For the

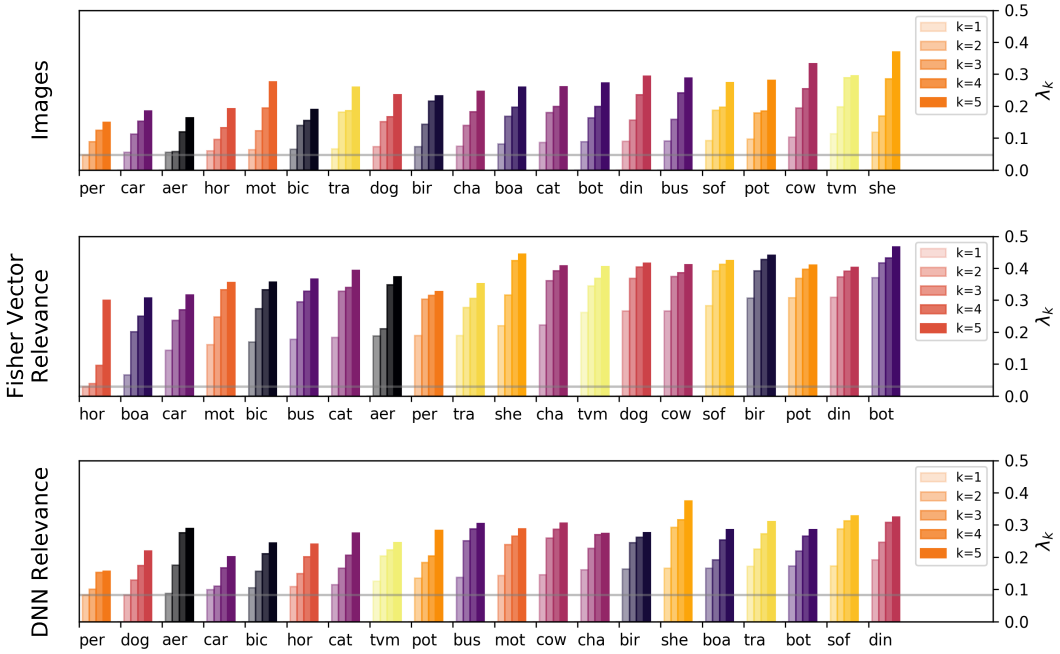

FIGURE 6.5: **The smallest 5 eigenvalues of L_{sym} for different input types and all Pascal VOC object categories.** The bar charts are colorized wrt to the alphabetical order of the class names and ordered from left to right wrt to λ_2.

FV model, the first two to three eigenvalues are comparatively low, compared to all other classes in all other settings. The eigenvalue spectra for the majority of classes behave similarly in all settings. That is, λ_2 is significantly larger than λ_1 (which is always 0 in all cases, as expected), with the consecutive distances between the following eigenvalues being approximately equal, which speaks for one single well-connected point cloud without disjoint clusters. When comparing λ_2 in all three settings, we observe its value is gradually rising between neighboring classes in the ranking order. Between the classes "boat" ranked 2nd and "car" ranked 3rd on the bar plot regarding the FV model, there is a noticeable gap, motivating closer inspection via Figure 6.6.

Image inputs for class "boat" are clustered due to visual similarity and do not show – except for one outlier image showing a sundown scene – unexpected information. The DNN embeddings, together with the relevance plots reveal the locally heterogeneous prediction structure of the DNN, which predicts based on the shape of the shown boats themselves and structures related to boats such as sails. The results for the FV model however reveal one to two weakly connected subclusters besides the main group of samples, with frequently reoccurring relevance structure especially related to true positive predictions (visualized via red aurae). The pattern the FV model has learned to predict on is the water around and below the boats, not the boats themselves. Closer inspection reveals that features from the boats themselves are rated as contradictory information by the FV classifier. We relate the strong reaction of the FV model to water patters to the spatial pyramid mapping scheme (Lazebnik et al., 2006; Bosch et al., 2007) used in the computation of the FV descriptor (Chatfield et al., 2011). The use of spatial pyramid mapping allows models to incorporate a weak sense of global shapes and scene geometry into the optimization process, and is known to improve the prediction performance among Bag of Words type models. Conversely, this method of feature pooling also facilitates the development of spatial biases, which might also have affected class "horse".

Another class raising further interest is the class "aeroplane". Here $\lambda_2 \approx \lambda_3$ for the input images already and for the DNN results show an uncharacteristically fast increase in eigenvalues (*i.e.* there are large gaps between several of the smallest eigenvalues), despite the class being ranked 3rd in Figure 6.5.

Figure 6.7 shows that the low 2nd and 3rd eigenvalues for images can be explained by a

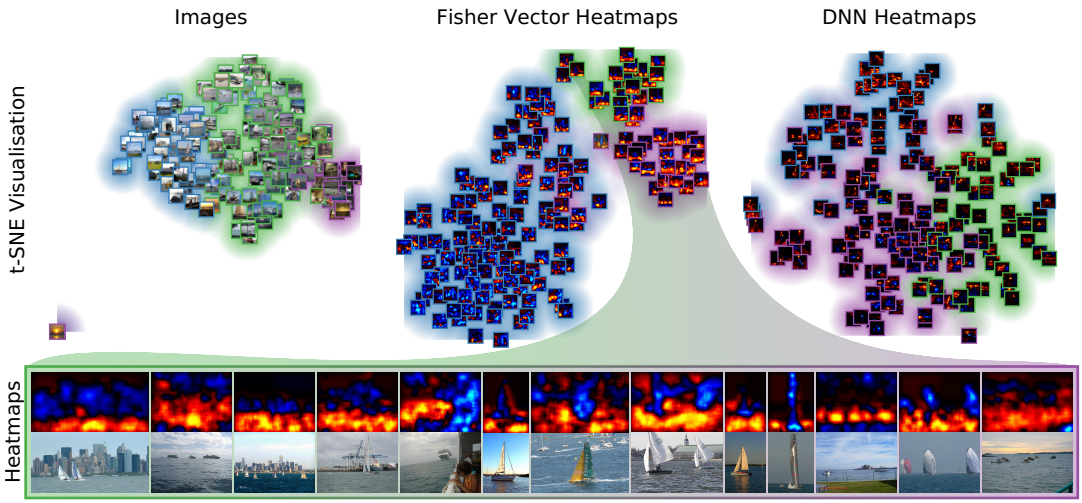

FIGURE 6.6: **Cluster label assigments for class "boat" via SC for input images, FV model relevance maps and DNN relevance maps.** Embedding coordinates in $\mathbb{R}^2$ for visualization have been computed on pair-wise distances derived from the weighted affinity matrix W used for SC. The samples at the bottom of the figure show FV relevance maps without preprocessing together with corresponding input images.

cluster of samples which t-SNE embeds as the lower left arc in the visualized $\mathbb{R}^2$-coordinates. The cluster contains images of airplanes in flight in front of blue clear sky. The other half of the embedded images show planes on the ground on rollways and in front of airport structures. The DNN relevance maps cluster a subset of images in front of blue sky, but for an entirely different reason, which becomes apparent by inspecting the relevance maps of the small cluster in the top right of Figure 6.7: Relevance maps reveal that the DNN model predicts these samples based on the top and bottom border of the image.

We assume the model has learned to predict based on image information introduced as part of the input preprocessing used for training the multi-label model for Chapter 4: For preparing the Pascal VOC images as inputs for our DNN model, we scaled down the images to 256 pixels on the longest edge and then padded the image into square shape by copying pixels from the image border, which is a common practice in computer vision. Then, the (227×227)-sized image center was cropped and used as input for the DNN model. See Appendix C.1 for choices made regarding input preprocessing steps for the DNN model.

The relevance maps show, that the DNN model picks up heavily on the border regions (especially top and bottom) introduced during image preprocessing, which reflects in the assigned cluster labels and embeddings in $\mathbb{R}^2$. Considering the DNN model's weak dependency on context for class "aeroplane" – as measured in Figure 4.6 in Chapter 4 – suggests that the border artefact detected by SpRAy is occurring consistently, even though the model does not dominantly depend on it for correctly recognizing airplanes.

We can also observe another artifact for the FV model's predictions which becomes apparent from the t-SNE at closer inspection but is not as apparent in the spectrum of eigenvalues. Again, with high probability due to the use of spatial pyramid mapping, the model reacts strongly to uniform and structureless background in the images, but dominantly in the top half of the image and especially in the top left quadrant. The model's reliance on a lack of structure in the background attributes negative relevance to any object occluding the background. The model has learned to generalize the class "uniform image background" instead of the intended class label "aeroplane". The results to further experiments supporting this conclusion can be found in Appendix E, Figures E.3 and E.4.

6.3.1 Verifying the Detected Bias in Prediction Behaviour

The results obtained with SpRAy in Figure 6.7 suggest a previously unknown bias in the decision making of the DNN model, namely that the classifier for class "aeroplane" (as part of

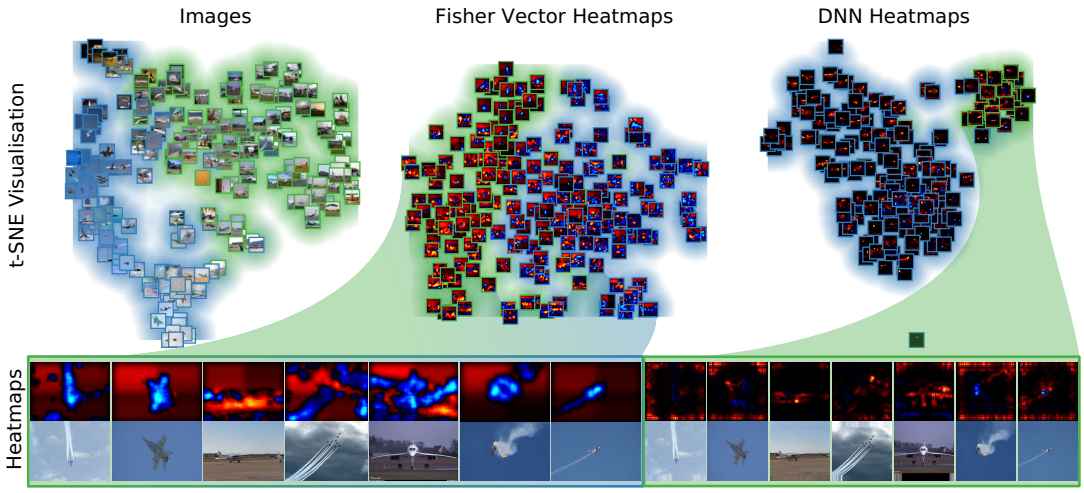

FIGURE 6.7: **Cluster label assigments for class "aeroplane" via SC for input images, FV model relevance maps and DNN relevance maps.** Embedding coordinates in $\mathbb{R}^2$ for visualization have been computed on pair-wise distances derived from the weighted affinity matrix W used for SC. The samples at the bottom right (square images) show DNN relevance maps and images with strong reaction of the DNN models to the border padding. FV relevance maps for the same images are shown to the left. Enlarged relevance maps and images are shown without preprocessing.

the trained multi-label DNN) has learned an image border devoid of structure as a characteristic describing the object class. To further our understanding and to verify our hypothesis of this undesired prediction bias (we wish for the DNN to predict based on the airplane itself) we compare different approaches for obtaining square images as DNN inputs from the Pascal VOC test samples, with examples shown in Figures 6.9 and 6.10: (1) Image padding by copying pixel values from the image border, as it is performed in the current image prediction pipeline. (2) Mirror padding, which extends the image as needed by copying image content and may add natural structures to the image border already present in the image. (3) Cropping of the largest possible square and centered image patch, which removes image content from the border and enlarges structures from the image center relatively. (4) Padding with random pixel values, adding non-uniform random structure of high frequency to the image border. (5) Padding with a random colour, uniformly applied to the padded image area. (6) Padding with sky-blue colour sampled from a test image, uniformly applied to the padded image area.

We hypothesize, that the model has learned to expect uniformly coloured image borders for samples belonging to class "aeroplane". We verify that hypothesis in Figure 6.8 (left), which shows that any processing resulting in a square image besides the copying of border pixels will result in a decrease *airplaneness* of the sample. Mirror padding can *e.g.* repeat the structure of clouds and cropping will altogether remove border information and move structures from the image center to the (new) image border.

Extending with random pixel values adds high frequency content to the image border, which directly contradicts what the model has learned, resulting in the highest measured decrease in the predictor output across all image processing dapproaches. This approach also reliably removes the artefact of positive prediction contribution of the image border as observable in Figure 6.9. While the overall prediction scores for noise-affected samples declines drastically, the center of the relevance maps, showing the actual images, remain almost the same.

Our experiment shows that padding the images with random yet per image constant colour value results in a lower decrease of the predictor output compared to random noise. This further verifies that the model expects the absence of structure-rich information. Setting the border colour to a sky-blue hue (as taken from test images) reduces the decrease of $f(\mathbf{x})$ to an lower extent than for cropping and a considerably lower extent than padding

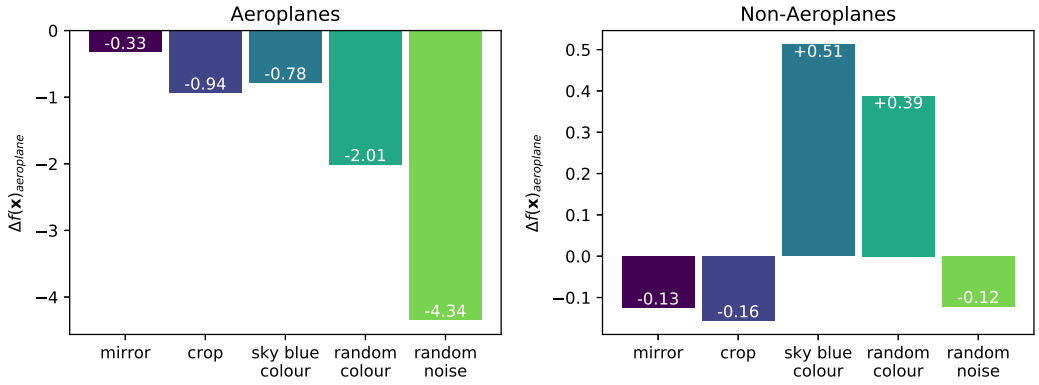

FIGURE 6.8: **Average change in predictor output with different image resizing strategies compared to border pixel copying.** *Left:* Averages over all "aeroplane" samples. *Right:* Averages over all samples *not* labelled as "aeroplane".

with randomly colours. For both examples in Figure 6.9, padding with sky-blue colour even (marginally) increases the predictor output compared to baseline (copy border) and enhances or even causes the strong appearance of the observed border artefact, despite not matching the *natural* image background colour. Further, we observe that the model reacts strongly to transitions beetween two uniformly coloured image areas for both padding variants adding uniformly coloured areas, as shown in Figure 6.9. This raises further questions about the influence of padding choices of the pre-training phase on ImageNet.

We conduct a second experiment to verify our hypothesis at hand of all test images *not* being labelled as "aeroplane" and apply the same padding approaches (See Figure 6.10 for examples). We provide those sample images as input to the DNN and observe the effect on the model output corresponding to the prediction of airplanes. The results are shown in Figure 6.8 (right) verify again that introducing structure (which is with a certain likelihood much stronger for non-"aeroplane" images) to the border regions of the image (mirror and crop, random noise) reduce the *airplaneness* of the inputs further, while padding with (a fitting) constant colour results in a comparatively high increase in the prediction score.

The relevance maps in Figure 6.7 show that positive relevance is strongest on the top and bottom borders of the image, while the left and right image borders only receive weak relevance attribution for aeroplane images. This classifier reaction makes sense intuitively, since photographing an object in the sky likely removes objects located on or near the ground (*e.g.* roof tops), due to the camera's tilt along the vertical axis.

We further investigate and subdivide all non-"aeroplane" images into images which require vertical padding (landscape format, $\approx$ 3800 images) and images which require horizontal padding (portrait format, $\approx$ 900 images). The results in Figure 6.11 show that padding the horizontal axis (Figure 6.11 (right)) has on average less of an effect to the increase of *airplaneness* via constant border padding compared to padding of the vertical axis (Figure 6.11 (left)).

We can conclude that the DNN has learned to detect airplanes in part based on uniform image areas of constant (sky-like) colour at the top and bottom of the image. Using that model as a predictor for airplanes outside of laboratory settings might cause high rates of false positive predictions when observing the sky with a camera system and feeding the image input to the model. With SpRAy, we have pin-pointed that previously unknown artefact in the model's prediction behaviour. Future iterations of the model, its training data and preprocessing choices could thus be adapted accordingly to avoid predictions of aeroplanes based on very low frequency image borders.

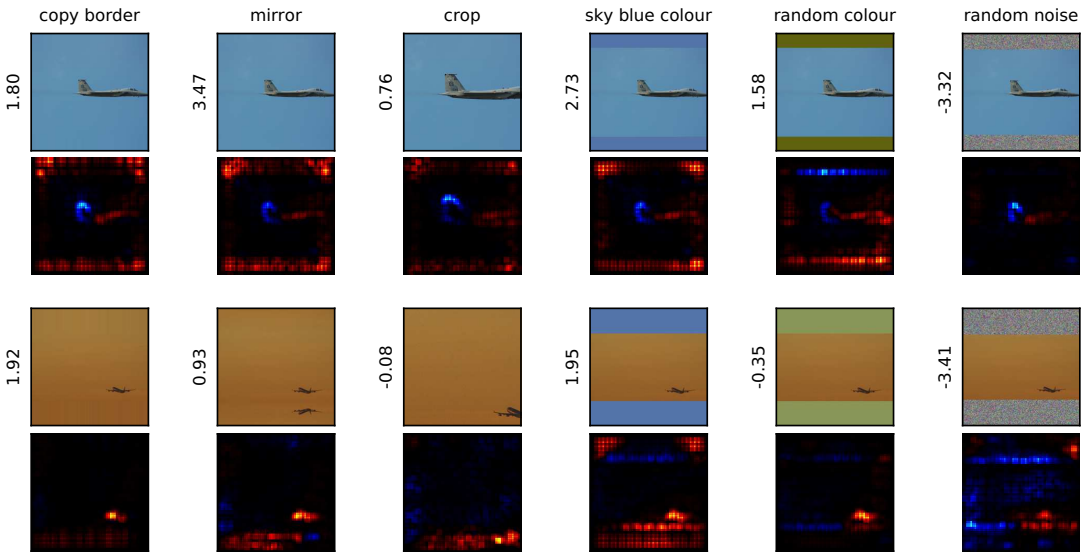

FIGURE 6.9: **Samples from class "aeroplane" and predicted scores for class "aeroplane", with corresponding relevance maps, as affected by different preprocessing strategies to obtain square images.** Padding with (high frequency) random noise effectively decreases the predictor output and removes the "border artefact". Using low frequency areas (of the right colour) for padding increases the predictor output for class "aeroplane" and may even introduce the "border artefact" in the first place.

6.4 Limitations

The Spectral Relevance Analysis method, as described in this chapter, has so far been used to analyze the models' prediction strategies based on relevance localization. In order to rigorously evaluate predictions of models based on features invariant to *e.g.* scale, shift and rotation, an extension of the algorithm might be required.

We present our results wrt to one set of parameters for the involved methods, as recommended in literature, which already provide some novel and interesting insights. However, it is not yet clear whether these parameters may be determined automatically – beyond brute-forcing the algorithm through a parameter grid – and how (in)sensitive the SpRAy method is to different parameter settings.

6.5 Conclusion

SpRAy enables researchers and machine learning engineers to understand the classifier's prediction strategy in detail and in only a matter of seconds, closing the gap between one-dimensional performance measures over test datasets – such as accuracy and loss ratings – and the manual assessment of explanation heatmaps for single predictions. With the help of relevance inputs generated with LRP, spectral analysis reveals the composition of selected sample sets in terms of different applied prediction strategies, *e.g.* whether a model's predictions are heterogeneous across all samples or show suspicious or artifactual behaviour, using unintended or surprising information. The complementary embeddings computed via t-SNE – a non-linear technique related to spectral clustering – aid in the presentation of the spectral analysis in a human-interpretable manner. Our work presents a comparative analysis of the eigenvalue spectra of all object categories of the Pascal VOC 2007 test set – a widely used benchmark dataset in computer vision – which points out issues with the composition of some classes on their own and in combination with choices made regarding the model architecture and preprocessing steps for both the investigated FV and the DNN predictor.

We found a systematic bias in the prediction strategies of the FV model for classes "boat", "horse" and "aeroplane" connected to the spatial pyramid mapping scheme employed to

FIGURE 6.10: **Non-"aeroplane" samples and predicted scores for class "aeroplane", as affected by different preprocessing strategies to obtain square images.** While the DNN model clearly recognizes parts of the shown non-aeroplane objects as contradicting evidence, the addition of uniformly colored image extensions provokes a positive response in the prediction and relevance map for class "aeroplane".

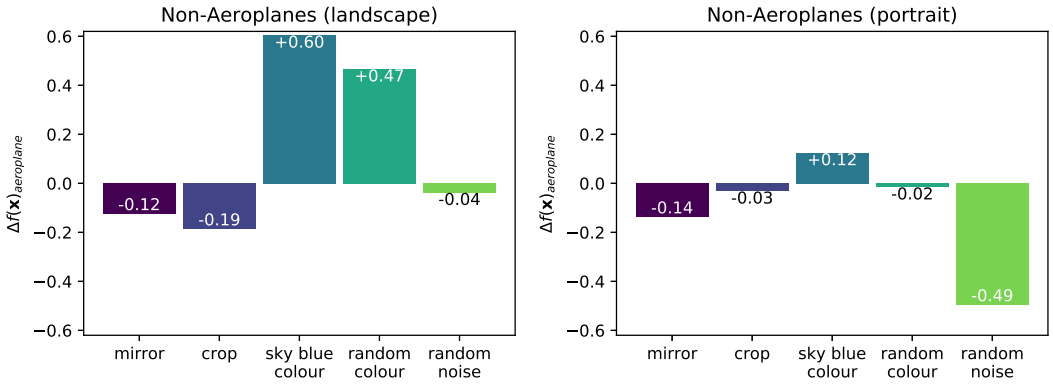

FIGURE 6.11: **Average change in predictor output with different image resizing strategies compared to border pixel copying for non-"aeroplane" samples.** *Left:* Landscape format samples, padded *vertically* (or cropped horizontally). *Right:* Portrait format samples, padded *horizontally* (or cropped vertically).

profit from global shape and geometry information. Regarding the DNN model, our pipeline has helped with the identification of a previously unknown issue caused by the preprocessing of the network inputs, which can thus be rectified in the model's next iteration in the development cycle. Our work proposes an initial application of relevance maps as features

for detailed model analysis, focussing on the strategies as applied by a trained model and as revealed by a meaningful set of explanation heatmaps.

For future work, we see much potential in the diverse applicability of SpRAy. The choice of explanation method is not limited to LRP and may be used to direct the semantics of the analysis. A first preprocessing choice has been a simple sum pooling algorithm for aggregating relevance onto a smaller grid of values, which preserves the locality of the relevance responses in a robust manner. Different choices for preprocessing will guide the focus onto specific aspects within the heatmap, *e.g.* incorporating an interpolation component could act as a low pass filter and revealing different structures in the explanation (see Appendix E for corresponding results). Applying Fourier transforms on the relevance maps in order to operate in the frequency domain instead of the pixel space is another promising alternative for preprocessing.

Chapter 7

Conclusion

We conclude this thesis with a summary of the presented methods and results, provide an overview over possible future research directions and unanswered questions, and give examples of later work profiting from analyses with Layer-wise Relevance Propagation.

7.1 Thesis Summary and Outlook

In this thesis, we have proposed a novel method – *Layer-wise Relevance Propagation* – for gaining insight into individual classifier predictions. LRP is motivated by the measurable and transparent feature contributions inherently available for linear methods, to which we refer to as *relevance*. LRP generalizes the computation of these quantities for models comprised of stacks of (non-linear) feature transformations, approached as decomposition of the predicting function, proportionally to forward contributions determining the function output. The method is primarily described in the broad context of predictors as systems of forward-directed activations, or *forward messages*. This leads to the applicability of LRP to a wide range of classifier types and architectures due to the reduced set of requirements for, and assumptions about the model to analyze, compared to other approaches for interpretable machine learning. This includes Deep Neural Networks, Support Vector Machines and Bag of Words feature extraction pipelines and also discontinuous and indifferentiable mapping functions.

In Chapter 3, a range of desiderata for interpretation methods have been discussed. LRP does fulfill those properties and compared to competing methods better identifies the input features considered as relevant by the model, as evaluated quantitatively. We show that relevance maps are not only meaningful, but also can be computed efficiently. This is true especially for Deep Networks, where the decompositions performed by LRP can be expressed as a modified gradient backward pass.

The analyses performed in this thesis are set in the domain of image recognition, due to the human-parsable, yet complex nature of image data. Using LRP on image recognition models of different complexity revealed that deeper and more complex models, which tend to generalize better, do so by more selectively using information from the input compared to shallow models. Shallow predictors, such as the analyzed Fisher vector SVM, tend to use much contextual input information as hypothesized in (Everingham et al., 2009); to a much higher extent compared to models of Deep Learning, due to design and the comparatively limited capability to filter out irrelevant information over additional layers. We conclude that heatmap sparsity is linked to the generalization performance of a model, which raises the question for future work, whether relevance information may be used as criterion of quality for the evaluation of models on unseen data lacking ground truth labels.

Our findings on the widely used Pascal VOC 2007 dataset suggest that the failure of a model to solve a task (humanly) meaningfully usually does not lie with the learning method itself. On the contrary, the learning machines performed admirably in the given setting: In Chapter 4 we found out that the evaluated (and among Bag of Words models state of the art) Fisher vector model predicted the label "horse" as well as an on average much more accurate Neural Network model, but for entirely different reasons. While the network used parts of the horse as cues for decision making, the Fisher vector model relied to large parts on the presence of a copyright watermark in the images, or hurdles in images taken in a competitive riding setting. This *failure* of the Fisher vector model – failure in terms of learned prediction

strategies not intended via the label description "horse" – can be explained with the choices made for a mapping function integral to the improved accuracy of the model on the test set, and the composition of the dataset itself: Firstly, a pyramid mapping scheme promoting the learning of weak geometrical scene understanding is part of the model; The hurdles and the watermark learned as representative features both are located in the bottom of the image, which makes it easy for the model to adapt to these characteristics. Secondly, roughly one fifth of all "horse" images contain the same copyright watermark. The affected images have been collected from a publicly available and riding-themed online repository. Combining these circumstances links the "misperformance" of the model to human error, not seldomly found in model design and dataset curation. Similar results are also obtained from the analyses of model predictions on the Adience benchmark dataset for age and gender recognition on images of faces.

With the steady progress of (deep) machine learning, it is clear that our here presented contributions can not exhaustively satisfy the requirement for transparency in machine learning in general: This thesis mainly covers sequential model architectures and proposed advanced and purposes decomposition rules, alongside recommendations for the application thereof. Further, all proposed backward rules consider mappings based on sum-, max-, or generalized p-means-aggregated activations. The work of (Arras et al., 2017b) proposes a solution for multiplicative neuron interactions in Recurrent Neural Networks (RNN) such as Long Short-Term Memory (LSTM) (Hochreiter et al., 1997) models and (S. Bach, 2013) outlines options for the decomposition of conditional probabilistic prediction functions. To our knowledge no proven and well justified work on LRP for more modern architectures such as highly interconnected ResNet architectures (He et al., 2016; Zagoruyko et al., 2016) (beyond an application of the decomposition rules proposed here), or fully multiplicative models such as Markov Random Fields does exist to this day, and is subject to future work. While Layer-wise Relevance Propagation, as described in this thesis, in principle also is applicable to regression models, a further extension of the decomposition framework to unsupervised methods or the family of *e.g.* (kernelized) clustering methods might yield interesting insights.

Since the manual screening of hundreds of individual relevance maps in order to better understand a model is a labourous and tiring task, the *Spectral Relevance Analysis* (SpRAy) method is proposed, building on Spectral Clustering and analysis of relevance maps. We have shown that SpRAy can be used to systematically screen large sets of explanatory relevance maps for common patterns in a model's reasoning, closing the gap between the reporting of test set wide average label recognition performances and detailed assessment of the reasoning of a model at hand of individual samples. A first application of SpRAy on the models operating on PascalVOC revealed additional and previously unknown systematic prediction artefacts for a Neural Network predictor, caused by a not entirely appropriate preprocessing of samples. While SpRAy constitutes a first step for using LRP towards feasible and more complex large scale analyses, future research towards improving or guiding the training of models via relevance information seems promising. More concretely, can relevance information be used to represent a *desired* way of decision making, *e.g.* in a teacher-student setting, to guide a learning system during training?

7.2 Impact

Layer-wise Relevance Propagation is a versatile method applicable to a wide range of model architectures, prediction problems and data domains. LRP has been used to decompose Deep and Convolutional Neural Network predictors, *i.e.* in the analyses presented throughout this thesis, and on numerous other occasions, *e.g.* in (Binder et al., 2016c; Arras et al., 2017a). Applications to classical (kernelized) Bag of Words prediction pipelines beyond the topics covered in this thesis can be found in (Srinivasan et al., 2017; Arras et al., 2017a; Binder et al., 2018). Work on explaining anomaly detectors can be found in (Schwenk et al., 2014; Kauffmann et al., 2018; Amarasinghe et al., 2018), with interpretability methods for clustering methods being in preparation.

Our method has generated a strong impact, especially in fields related to deep learning, and has already seen application in various settings in the recent past. In the following,

we will provide an exemplary overview over existing applications of LRP in the sciences, ordered by application domain.

7.2.1 Applications in Image Recognition

An early test bed for the application of LRP has been the MNIST (LeCun, 1998) hand written digit dataset (S. Bach et al., 2015). Images in general provide for an accessible and easily human-interpretable environment to demonstrate the method's capabilities in. The work of (Arbabzadah et al., 2016) applies LRP in a multifaceted prediction setting on face images with prediction targets such as age, gender, attractiveness or happiness. The authors use relevance feedback to investigate characteristic features responsible for predictions wrt othorgonal tasks, and compare the expressiveness of different output schemata for the analyzed networks, *e.g.* binary outputs vs multilabel outputs. (Seibold et al., 2018) studies the security-relevant task of detecting face-morphing attacks in automated identity verification. The paper uses LRP to gain insights into the (differently) trained models' decision making when under attack.

Further applications of LRP in image recognition include uses for *e.g.* weakly supervized object localization (Zhu et al., 2017; Fan et al., 2017), comparing feature imporance between trained DNNs and humans (Linsley et al., 2017) and deriving relationships between thematically similar images as identified by a DNN-based recommender system (Bharadhwaj, 2018).

7.2.2 Applications in Text Processing

A series of text processing applications adopting and extending LRP is (Arras et al., 2016; Arras et al., 2017a; Arras et al., 2017b; Horn et al., 2017). (Arras et al., 2017a) introduces LRP to the domain of Natural Language Processing (NLP) by comparing the explainability of CNN-based and Bag of Words-based text topic classificators. The paper demonstrates the superior relevant word identification performance of LRP (compared to SA) and introduces novel, semantically regular word representations based on relevance attributions. Relevance-based document summary vectors (gained from the CNN model) are shown to lead to better topic classification performance than SA-based vectors or classical TFIDF weighting schemes. The applicability of LRP is extended to RNN architectures in (Arras et al., 2017b), enabling the decomposition of multiplicative (gating) interaction within the network. The authors use a bi-directional LSTM model for sentiment analysis. Findings via relevance decomposition indicate that the LSTM's sentiment understanding is not static but highly contextual. Relevance information is used in (Horn et al., 2017) next to TFIDF-rankings for computing clouds of relevant words for quick dataset exploration and understanding.

Further applications in NLP and text processing include the use as a tool for analyzing the sentence production dynamics in generative RNNs (Calvillo et al., 2018), understanding end-to-end neural machine translation (NMT) (Ding et al., 2017) and providing explanations in text-based stock prediction (Shi et al., 2018) and document classification (Pörner et al., 2018).

7.2.3 Applications in Audio and Video Processing

An application of human action recognition in compressed domain is presented in (Srinivasan et al., 2017). With the help of LRP the authors explain the decision of a Fisher vector based Bag of Words predictor on compressed video data from the HMDB51 dataset (Kuehne et al., 2011). Relevance information shows that the model predicts an action shown in the video, based on only a few, significant frames. For the first time, (Anders et al., 2018) explains CNNs trained for human action recognition tasks on the Sports1M dataset (Karpathy et al., 2014b). Relevance maps computed with the more strictly formulated LRP variant DTD reveals that the CNN is successfully able to identify the (for us humans) class-relevant information in the video. The paper analyzes patch- or snippet-based training, which is used in practice to reduce the number of free parameters of the model under the (often violated) assumption of locality of label-relevant information. The authors describe an observed "border effect" in the prediction explanation, corresponding to a temporal looking-ahead of the

network to later frames in the video. Comparisons of the modi operandi between DNNs operating on audio signals – either in spectrogram form or as raw waveform data – are performed in (Becker et al., 2018) on a newly created spoken digit dataset. Reported findings indicate that the spectrogram-fed networks seem to use meaningful spectral bands of the input data for decision making. The model predicting on raw waveform data, however, was found to only use the outer hull of the signal (*i.e.* the relatively small fraction of samples of locally maximal amplitude), which provides an explanation of the noticable gap in prediction performance to the spectrogram-fed networks on gender and digit recognition tasks.

Further applications of LRP in the audio domain include *e.g.* RNN-based audio source localization (Perotin et al., 2018), and a comparative study investigating relevant features in human speech recognition (HSR) and automated speech recognition (ASR) with DNNs (Spille et al., 2017), concluding that similar cues are inporant in ASR and HSR.

7.2.4 Applications in the Sciences

(Sturm et al., 2016) presents a first use of LRP for explaining the decisions of DNN classifiers in cognitive neuroscience, where the potential of deep learning methods hat not yet been fully explored due to these models' black box nature. LRP heatmaps reveal neurophysiologically plausible patterns learned by the DNN. While the scalp map patterns derived with CSP (Blankertz et al., 2008) – a standard technique in Brain Computer Interfacing – represent aggregated information per class only, relevance maps are able to identify neural patterns in single time points of single trails. In the domain of human gait recognition, interpretable linear predictors are the method of choice (Phinyomark et al., 2017). The work of (Horst et al., 2018) shows with LRP that non-linear predictors learn physiologically meaningful features for subject prediction which align with expected features used by linear models. The paper shows that interpretability does not need to be a trade-off for model expressiveness, and demonstrates the increased robustness of DNNs to noise compared to less complex models, whithout having to sacrifice decision transparency. Another application in cognitive neuroscience is presented with (Thomas et al., 2018), which for the first time apply LSTMs to whole-brain fMRI data. The presented method outperforms conventional decoding methods. An adaption of LRP maintains interpretability and verifies the model's predictions are based on physiologically appropriate brain areas for the classified cognitive states. Another study using LRP for the localization of activating brain regions is (Gotsopoulos et al., 2018). For the task of (cancer) cell prediction from histopathology imagery, (Binder et al., 2018) uses a Bag of Words prediction pipeline due to its invariances to rotation, shift and scale of the input data. For the verification of the prediction results relevance maps are computed, offering per-pixel likelihoods which indicate the presence of tumorous structures. LRP is also used for the identification, localization and counting of cells, *i.e.* lymphocytes, in (Klauschen et al., 2018). The work of (Y. Yang et al., 2018) goes beyond the use of simple models and applies LSTM predictors together with LRP for transparent therapy prediction on patients suffering from metastatic breast cancer. Clinical experts verify that the features used for prediction as revealed via LRP largely agree with established clinical guidelines and knowledge.

In further work in the sciences LRP is used for the detection of lesions (Quellec et al., 2017) in diabetic retinopathy data (Kauppi et al., 2007), understanding of activity prediction across chromosomes (Kelley et al., 2018), for distinguishing between valid prediction candidates and artefactual predictions in supernovae detection (Reyes et al., 2018) and for understanding automated decisions on behavioural biometrics (Chong et al., 2018). In (Bojarski et al., 2018) our method is used as a qualitative baseline in the context of autonomous driving. An extended version of LRP called CLRP[1] is used to visualize how CNNs interpret individual protein-ligand complexes in molecular modeling (Hochuli et al., 2018).

7.3 Concluding

Contemporary research literature shows that present day machine learning provides very capable problem solving tools, which are used to build models based on only finite training datasets. Throughout this thesis (Chapters 4 to 6) and later work (*e.g.* (Arras et al., 2017a;

[1] The C stands for "conservative" and hints at a different approach to handling zero-activated neurons.

Becker et al., 2018; Lapuschkin et al., 2018)), relevance information computed via LRP has revealed cases where well-performing machine learning models behaved in (for humans) unexpected ways during inference. The reasons for these cases have been found in the composition of (man-made) datasets and choices made for data representation, which both might not always illustrate the intended application case sufficiently well.

Understanding of the *hows* and *whys* of a predictor's reasoning should therefore be considered an integral part of the assessment of model quality. With Layer-wise Relevane Propagation, we provide a tool which can not only be used to understand machine learning models (Arbabzadah et al., 2016; Sturm et al., 2016; Ding et al., 2017; Linsley et al., 2017; Spille et al., 2017; Srinivasan et al., 2017; Anders et al., 2018; Binder et al., 2018; Calvillo et al., 2018; Chong et al., 2018; Gotsopoulos et al., 2018; Hochuli et al., 2018; Horst et al., 2018; Klauschen et al., 2018; Pörner et al., 2018; Shi et al., 2018; Thomas et al., 2018; Y. Yang et al., 2018), but also for building improved and *trustworthy* models based on the insights gained from relevance maps – *e.g.* via the derivation of better preprocessing steps, feature representations and appropriate augmentation of flawed training datasets (Arras et al., 2017a; Horn et al., 2017; Becker et al., 2018; Lapuschkin et al., 2018; Reyes et al., 2018) – or for potentially generating valuable domain knowledge from the reasoning of the machine (Lapuschkin et al., 2018).

Appendix A

Decomposing Non-Linear Classification Functions with Layer-wise Relevance Propagation

A.1 LRP for Feature Extractor Pipelines and SVMs

Algorithm 2 describes an algorithmic run-down of LRP for Bag of Words classification pipelines with Support Vector Predictors. Also compare Algorithm 2 to the basic algorithmic formulation of LRP in Algorithm 1.

Section A.1.1 presents example relevance decomposition for SVM classifiers based on various commonly used kernel types, as well as kernel-based anomaly detectors.

Algorithm 2: LRP for Bag of Words features with SVM classifiers

Data: Image $I = \{p\}$, Local descriptors $L = \{l\}$, BoW feature $\boldsymbol{x}$
model and mapping parameters
optional: Taylor root point $\widetilde{\boldsymbol{x}}$
Result: $\forall p \in I : R_p^{(1)}$

$R_f^{(4)} = f(\boldsymbol{x})$;
for $d \in 1 \dots D$ **do**
  $R_d^{(3)}, R_b^{(3)}$ as in Equation (2.72) or (2.68);
  optional: $R_d^{(3)}$ as in Equation (2.73) or (2.74);
end
for $l \in L$ **do**
  for $d \in 1 \dots D$ **do**
    $R_{l \leftarrow d}^{(2,3)}$ as in Equation (2.78) or (2.79)
  end
  $R_l^{(2)} = \sum_{d=1}^{D} R_{l \leftarrow d}^{(2,3)}$
end
for $p \in I$ **do**
  $R_p^{(1)}$ as in Equation (2.85)
end

A.1.1 Example Decompositions for Kernelized SVMs

Linear Kernel SVM

The decomposition of a SVM classifier based on linear kernel transformations is trivial, since the learned vector $\boldsymbol{w}$ weighting all input dimensions is accessible in data space. The representer theorem (Wahba, 1990; Schölkopf et al., 2001) states that a solution to a quadratic problem can always be found as a linear combination of the training data in the reproducing kernel Hilbert space: $\boldsymbol{w} = \sum_i \alpha_i y_i \Phi(\boldsymbol{x}_i)$. In case of the linear kernel, the mapping Φ

from data space into the kernel feature space is the identity function. A decomposition into relevance values for each input dimension d and the bias b is thus given as

$$R_d = x_d w_d \; ; \; R_b = b \; , \tag{A.1}$$

which is identical to the solution obtained when considering the dual formulation of the SVM prediction function and substituting k in Equation (2.72) with the linear kernel.

Histogram Intersection Kernel SVM

The Histogram Intersection Kernel (Swain et al., 1991; Maji et al., 2008) measures the intersection of histogram type input features, by projecting the kernel inputs as binary vectors into a feature space, of which the dimensionality depends on the largest counts of all histogram bins of all features. Its corresponding kernel function

$$k_{HI}(\mathbf{x}, \mathbf{y}) = \sum_d \min(x_d, y_d) \tag{A.2}$$

already is sum-decomposable over dimensions d in input space. Via Equation (2.72) we directly obtain

$$R_d = \sum_{i=1}^{S} \alpha_i y_i \min(x_{i(d)}, x_{(d)}) \tag{A.3}$$

Gaussian RBF Kernel SVM

Due to the exponential distance mapping of the Gaussian RBF kernel and its infinite-dimensional kernel feature space (Steinwart et al., 2006), a decomposition into contributions for all input dimensions can not be obtained directly. Instead, Taylor decomposition as in Equation (2.76) can be applied. The Gaussian RBF kernel is defined as

$$k_{GRBF}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|_2^2}{\sigma^2}\right) \tag{A.4}$$

where σ is the kernel width. Plugging its partial derivative wrt the input sample's components into Formula (2.76) yields the relevance decomposition formula

$$R_d = (\mathbf{x} - \widetilde{\mathbf{x}})_{(d)} \cdot \sum_{i=1}^{S} \alpha_i y_i k_{GRBF}(\mathbf{x}_i, \widetilde{\mathbf{x}}) \cdot \left(\frac{2x_{i(d)} - 2\widetilde{x}_{(d)}}{\sigma^2}\right) \tag{A.5}$$

χ^2 Kernel SVM

The χ^2 kernel function is especially popular in image recognition due to its good performance (J. Zhang et al., 2007; Vedaldi et al., 2009; Binder et al., 2013), but also can not be decomposed directly due to the exponential mapping surrounding an otherwise component-wise evaluated function

$$k_{\chi^2}(\mathbf{x}, \mathbf{y}) = \exp\left(-\sigma \sum_d \frac{(x_{(d)} - y_{(d)})^2}{(x_{(d)} + y_{(d)})}\right) \tag{A.6}$$

with $0/0 = 0$ in case of $x_{(d)} = y_{(d)} = 0$. Computing and inserting the kernel function's derivation into Equation (2.76) results in an approximated relevance contribution

$$R_d = (\mathbf{x} - \widetilde{\mathbf{x}})_{(d)} \cdot \sum_{i=1}^{S} \alpha_i y_i k_{\chi^2}(\mathbf{x}_i, \widetilde{\mathbf{x}}) \cdot \sigma \cdot \left(\frac{4(x_{i(d)})^2}{(x_{i(d)} + \widetilde{x}_{(d)})^2} - 1\right) \; . \tag{A.7}$$

Decomposing Kernel-based Anomaly Detectors

Both the One Class SVM (OCSVM) (Schölkopf et al., 1999) and Support Vector Data Description (SVDD) (Tax et al., 2004; W.-C. Chang et al., 2013) are unsupervised learning machines predicting in kernel space. As such, the prediction functions of both methods are similar in structure to the prediction function of the SVM, albeit the semantics of the predictions (and relevance decompositions) being deviant. (Schwenk et al., 2014) have shown that the rules defining LRP (*i.e.* proportional decomposition) can also be applied to both predictors trivially, in case the kernel functions are decomposable into sums of component-wise evaluations.

The prediction function of the OCSVM is

$$f(\mathbf{x}) = \sum_{i=1}^{S} \alpha_i k(\mathbf{x}_i, \mathbf{x}) - b \tag{A.8}$$

The OCSVM learns to separate its (unlabelled) training set from the origin of the coordinate system in feature space. The weighted sum of kernel functions over support vectors $\mathbf{x}_i$ in above equation measures the projected distance of a test sample $\mathbf{x}$ to the origin of the coordinate system in feature space, while the bias b is a learned threshold dividing *normal* samples ($f(\mathbf{x}) \geqslant 0$) from anomalous samples. For kernels matching the pattern of sum-decomposability in Equation (2.72), a relevance decomposition for input each input component d can be achieved easily as

$$R_d = \sum_{i=1}^{S} \alpha_i k(\mathbf{x}_{i(d)}, \mathbf{x}_{(d)}) \tag{A.9}$$

where each R_d describes a component of the input sample in terms of *normality* or *inlierness*. A recent study (Kauffmann et al., 2018) expands on that approach and presents solutions for interpreting OCSVM predictions with differentiable and distance-based kernel functions by performing Deep Taylor Decomposition (Montavon et al., 2016; Montavon et al., 2017).

The SVDD fits the smallest possible sphere enclosing the training data, as defined by a learned centroid $\widehat{\mu}$ and a radius $\widehat{r}$. Its dual prediction function is

$$f(\mathbf{x}) = \widehat{r} - a(\mathbf{x}) \tag{A.10}$$

$$a(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - 2\sum_{i=1}^{S} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + \sum_{i=1}^{S}\sum_{j=1}^{S} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{A.11}$$

where the term $a(\mathbf{x})$ is the squared euclidean distance between $\widehat{\mu}$ and $\mathbf{x}$ in kernel space. A prediction point is considered anomalous if $f(\mathbf{x}) < 0$. Inverting the sign of the prediction function, such that

$$f(\mathbf{x}) = a(\mathbf{x}) - \widehat{r} \tag{A.12}$$

results in an equivalently expressive predictor which detects inputs $\mathbf{x}$ as anomalous if $f(\mathbf{x}) > 0$. A decomposition of the inverted prediction function in Equation (A.12) yields a much more human-interpretable meaning of relevance

$$R_d = k(x_{(d)}, x_{(d)}) - 2\sum_{i=1}^{S} \alpha_i k(x_{i(d)}, \mathbf{x}_{(d)}) + \sum_{i=1}^{S}\sum_{j=1}^{S} \alpha_i \alpha_j k(x_{i(d)}, x_{j(d)}) \tag{A.13}$$

as a rating of *anomaly* or *outlierness* for each input component d.

Appendix B

Evaluating and Understanding Heatmaps

B.1 Additional Perturbation Experiments

We performed the perturbation experiments from Section 3.5 also with region sizes smaller and larger than (9×9), namely (1×1), (3×3), (5×5) and (19×19). Figure B.1 shows the results relative to the Sensitivity-ℓ_2 baseline. Due to the high computational load we did not compute the random baseline and also left out the ℓ_∞ norm as it performed very similarly to ℓ_2. Since $\text{LRP}_{\epsilon=100}$ performed much better than $\text{LRP}_{\epsilon=0.01}$ in Section 3.5, we decided to leave out the latter and to include an LRP variant with an ϵ value in between 0.01 and 100. We decided to take $\epsilon = 1$.

Qualitatively the results are very similar to the one reported in Figure 3.5 for the (9×9) region size. In all cases the ϵ-stabilized LRP algorithm provides best explanations. Interestingly, $\epsilon = 1$ outperforms the results obtained with significantly larger or smaller stabilizing factors. Thus although stabilization is important as discussed in Section 2.2.6, too large ϵ values may decrease the efficiency of LRP as it suppresses contributions from small relevances. Another observation which can be made from Figure B.1 is that the Deconvolution method outperforms Sensitivity Analysis in terms of AOPC only for large region sizes ((9×9) and (19×19) in Figure B.1. Note that the explanations computed by Deconvolution show a large number of responses that look like filter visualizations (see Figure 3.6 and Figures B.2, B.3 and B.4). The kernel size for the used networks is 11. We argue that with a region size close to or larger than the kernel size, we are obtaining a score of the region that is close to the score of the filter as a whole. Perturbing such a region has a large impact on the filter response and thus leads to a decrease of the classification score. Destroying individual pixels (or small regions) on the other hand only slightly affects the filter response and results in small AOPC values. LRP performs well for all region sizes, because it does not focus on filter responses but identifies the relevant pixels in the image. Note that these pixels often represent the shape of the object (see Figure 3.6 and Figures B.2, B.3 and B.4), so that destroying small regions around these pixels largely destroys the object shape which leads to a fast decrease of classification score.

B.2 Additional Heatmaps for Qualitative Comparison

We present additional heatmaps computed with Sensitivity Analysis, the Deconvolution Method and Layer-wise Relevance Propagation for comparison on the SUN397 dataset, the ImageNet dataset (ILSVRC2012) and the MIT Places dataset in Figures B.2, B.3 and B.4.

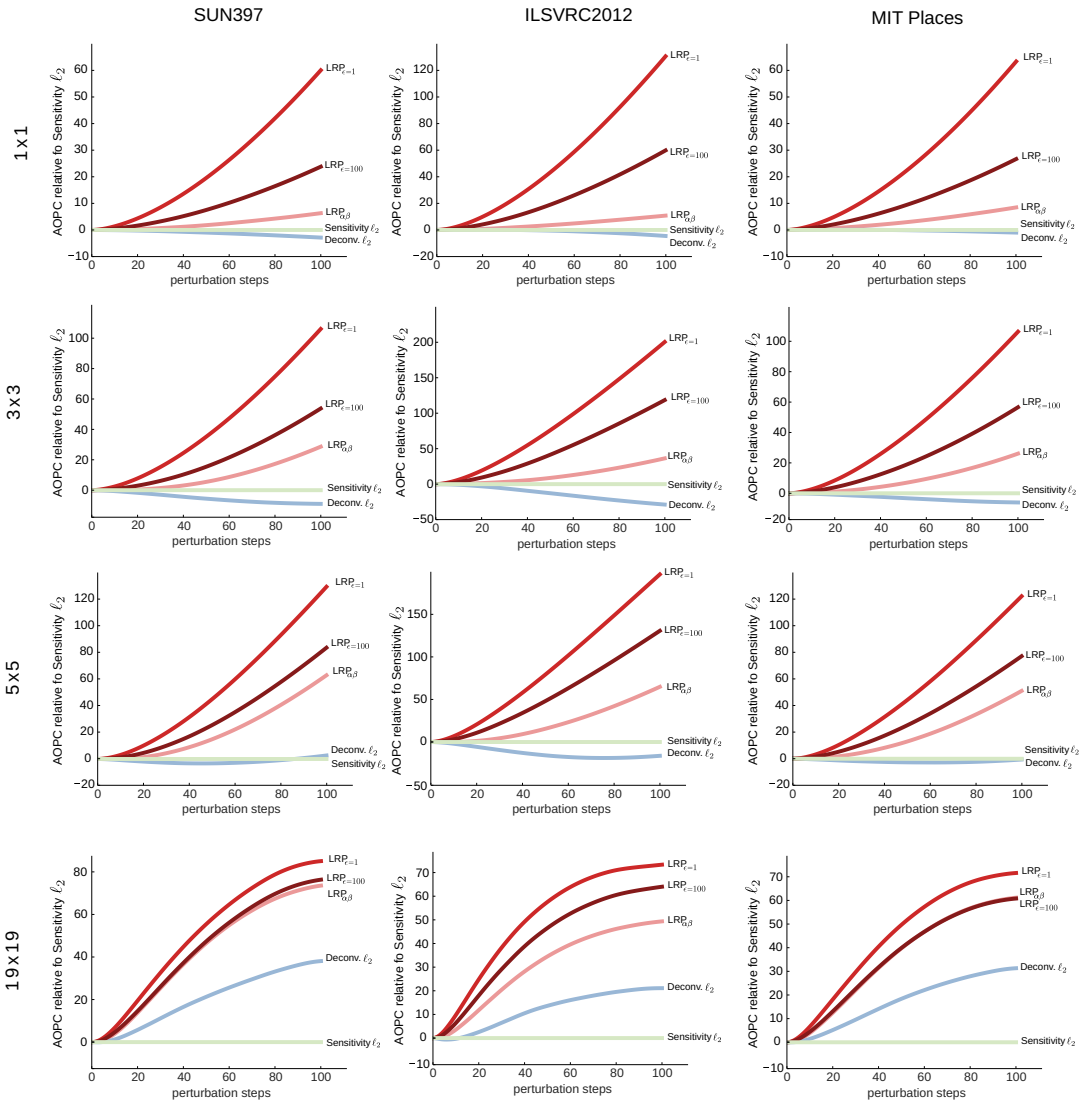

FIGURE B.1: **Comparison of the Deconvolution and LRP methods, relative to the Sensitivity ℓ_2 baseline for different region sizes.**

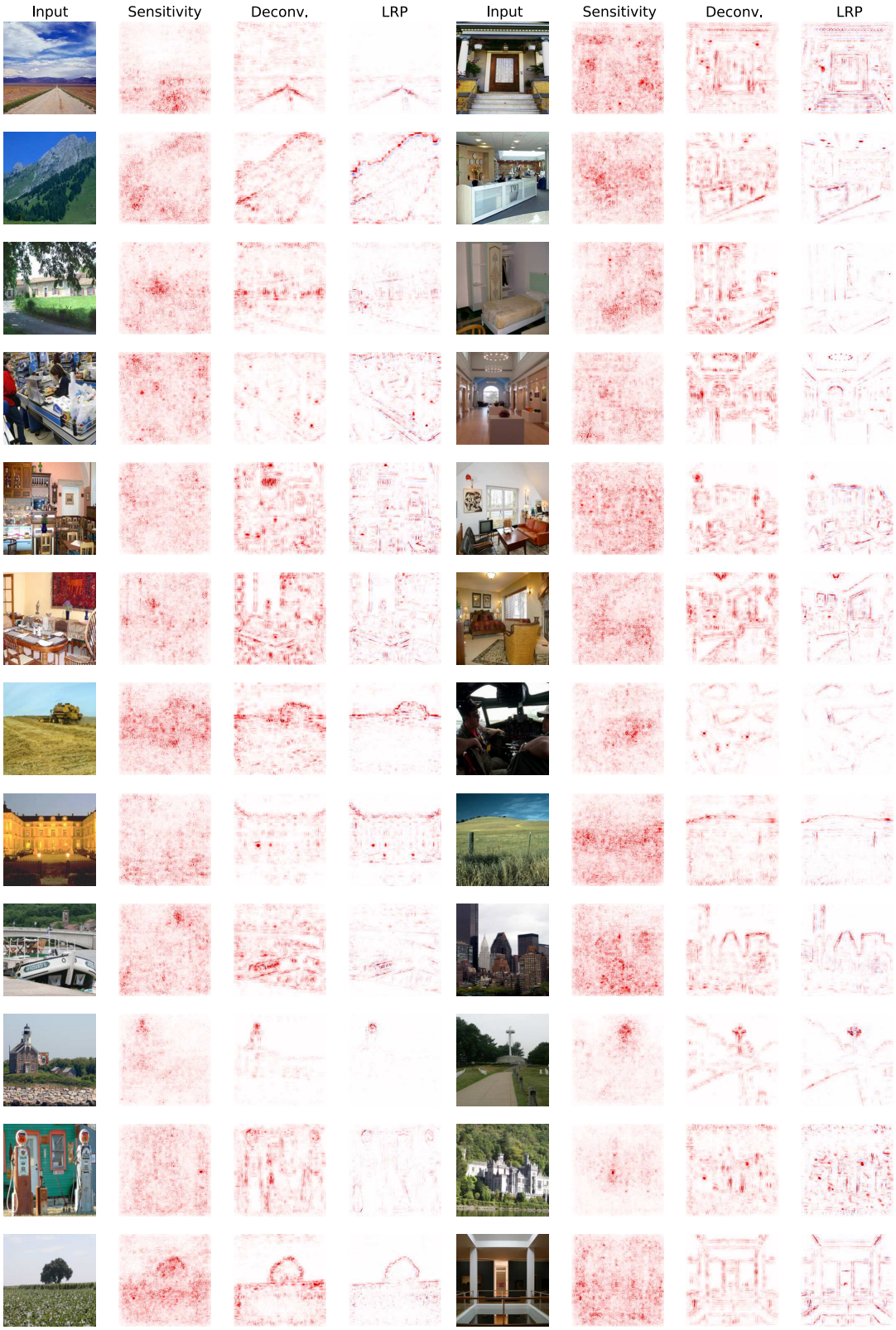

FIGURE B.2: **Qualitative comparison of the Sensitivity Analysis, Deconvolution and LRP methods for four exemplary images of the SUN397 dataset.** Red color indicates large scores, blue color indicates negative scores (LRP only).

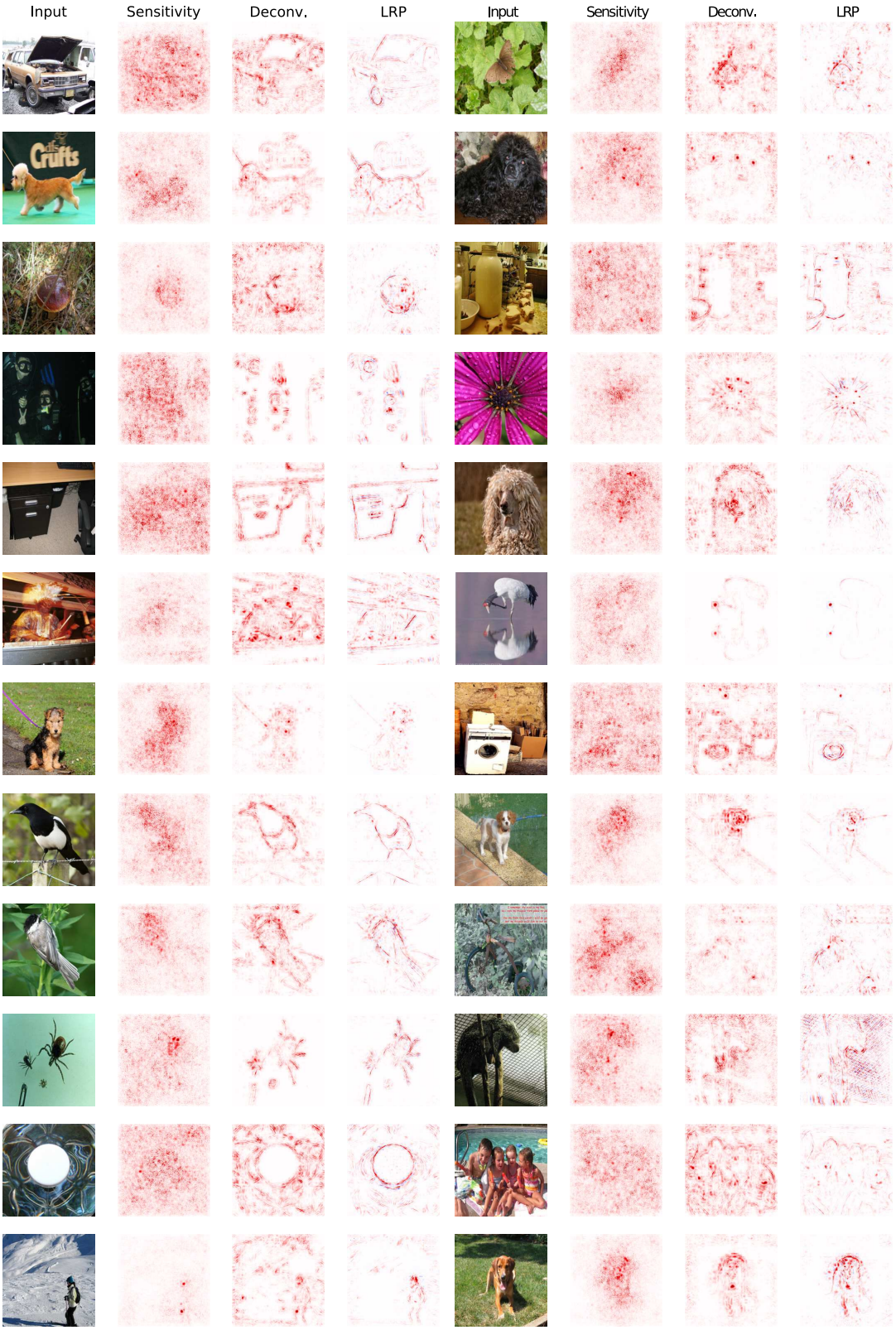

FIGURE B.3: **Qualitative comparison of the Sensitivity Analysis, Deconvolution and LRP methods for four exemplary images of the ILSVRC2012 dataset.** Red color indicates large scores, blue color indicates negative scores (LRP only).

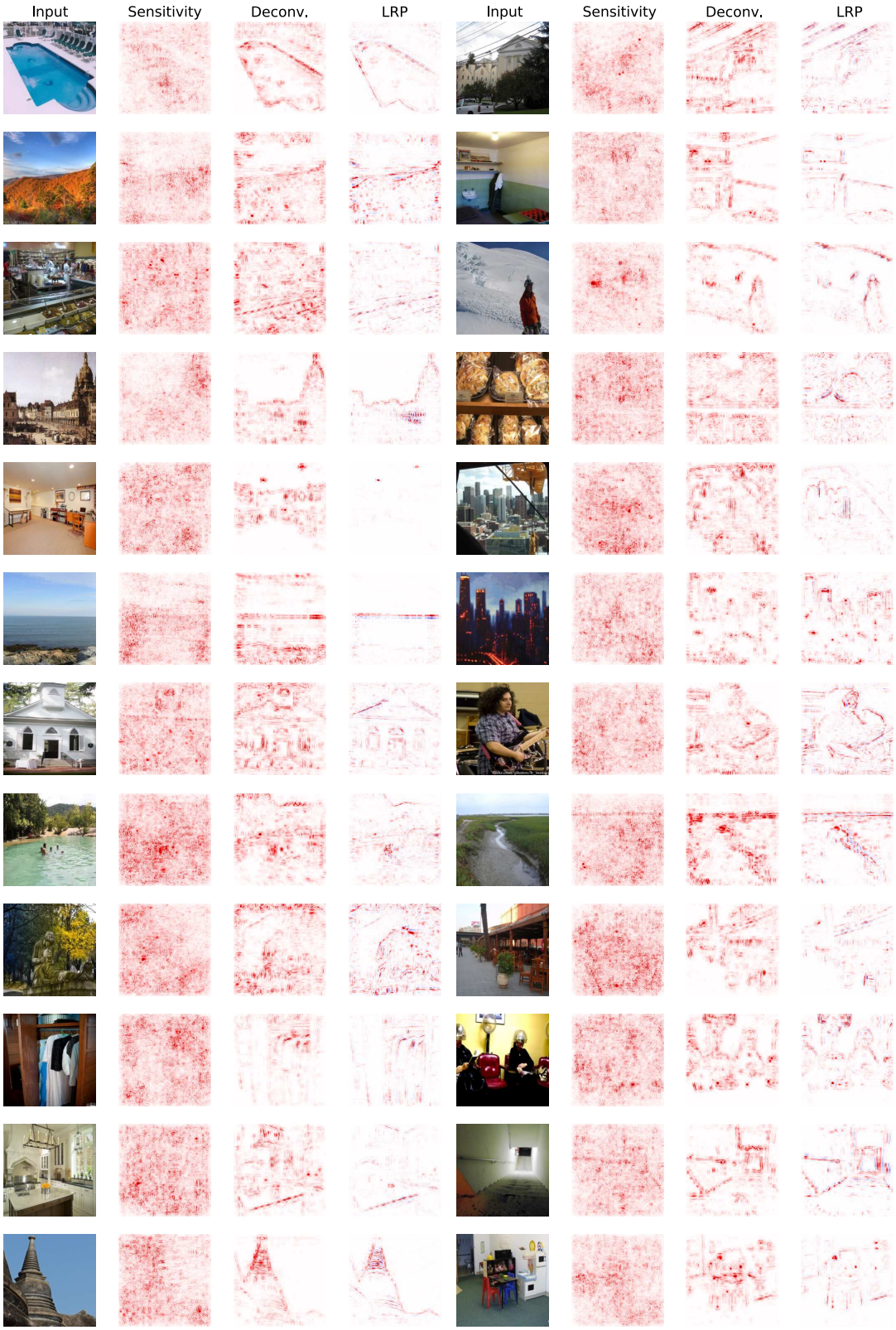

FIGURE B.4: **Qualitative comparison of the Sensitivity Analysis, Deconvolution and LRP methods for four exemplary images of the MIT Places dataset.** Red color indicates large scores, blue color indicates negative scores (LRP only).

Appendix C

Comparing Fisher Vector SVMs and Deep Neural Networks

C.1 Details on Neural Network Finetuning on the Pascal VOC Multilabel Setting

The starting point for the model was the BVLC reference CaffeNet net, as provided with the Caffe package (Jia et al., 2014). The training mode was a multi-label training, instead of the usual competitive multi-class training, because for PASCAL VOC multiple classes can be present in one image. The training criterion was the sum of hinge losses over all 20 classes in the Pascal VOC data set. Note that this required to use a customized image data layer and a customized hinge loss layer.

One general problem for training and testing is the question how to score an image and what data to use for training. A second problem is how to generate patches matching the square receptive field size from non-quadratic images. One general approach to generate non-quadratic images is to ignore the aspect ratio and to use warping in order to transform a non-quadratic patch into a quadratic one such as in (Girshick et al., 2014).

In order to have maximal comparability to Fisher vectors, which do not use warping and process an image as a whole, we decided for a simpler setup which is close to the setup used by Fisher vectors. This setup preserved the aspect ratio of patches used during training and testing, irrespective of the fact that other setups may have resulted in somewhat higher performance of the Neural Network.

The model weights are available as part of the Caffe Model Zoo[1].

C.1.1 Training Data Preparation

As we were interested in training a setup such that the Neural Network is able to use context, we refrained from training the network with image patches around the scale of a bounding box and smaller. We decided not to use the information about object bounding boxes for generating training data, because for Fisher Vectors this information was also not used for generating training data. Instead, each image was rescaled such that the largest side measured 256 pixels in length. The smaller side was padded at its boundaries by repeating the nearest pixel. From this modified image 4 edge and one center crop was taken. After mirroring the image, this was repeated. This resulted in 10 images per training image. This is a compromise to ensure a sufficiently large sample size for retraining, as it is known that Neural Networks excel typically at higher training sample sizes.

C.1.2 Test Data Preparation

Relevance maps were computed using one center crop only.

As for measurement of mean average precision, results depend on how to score one image at test time. Note that it is common to compute an average score over many crops of the image.

[1] https://github.com/BVLC/caffe/wiki/Model-Zoo#pascal-voc-2012-multilabel-classification-model

Resizing the largest side of the image to 256 pixels and using the (227×227) center crop only for each image resulted in the 72.12 mAP reported in Chapter 4, Section 4.5. Note that this setup corresponded to the setup used for computing relevance maps, so this was used for the sake of comparability.

Using a different test strategy, namely resizing the smallest side of the image to 256 pixels, then computing an average over a sliding window with stride of 20 pixels, resulted in an increase of ranking performance to 75.9 mAP. This strategy used merely 12 - 35 test patches per image. Using approaches with several hundred test windows, such as 500 windows in (Oquab et al., 2014) would probably have resulted in much higher mAP scores, however we did not consider a higher score relevant for the main message of the chapter, which is focused on context usage. In view of the considerably increased computation time for such approaches we refrained from them.

C.2 Computing Fisher Vector Perturbations by Replacing Local Descriptors

Chapter 3 describes in Equation (3.6) a heatmap evaluation method based in the step-wise perturbation of the model input $\mathbf{x}$ after an ordering $\mathcal{O}$, determined by a heatmap computed for each pixel of the input $\mathbf{x}$. In Chapter 3, this perturbation approach has been used to quantitatively compare the heatmaps computed with Sensitivity Analysis (Simonyan et al., 2013), Deconvolution Neural Networks (Zeiler et al., 2014) and LRP for Deep Neural Networks. Given the set of variants for computing decompositions from Chapter 2, Section 2.2.6, we specify the perturbation algorithm in Equation (3.6), aiming to identify the LRP decomposition variant best representing the predictions of the FV model (see Section 4.5.1) for use in further experiments (see Sections 4.5.2 and following).

The perturbation method has so far been applied to (the inputs of) Deep Neural Network models by switching the (almost) binary state of pixels of (28×28)-sized MNIST (LeCun, 1998) and by replacing image regions with random color values for up to (227×227)-sized color images.

For Bag of Words models such as the FV classifier evaluated in this chapter, perturbation strategies based on pixel replacement are less trivial to implement and evaluate than for DNN models, due to the dense extraction of local features l as a spatially orderless set L: (1) While replacing a single pixel in an image might not have a noticable effect on the classifier output, the associated process computing $\mathbf{x}^{(i)}_{\text{MoRF}}$ is computationally costly in comparison. From an image (resized to maximally (480×480) pixels) of the Pascal VOC dataset, on average 69,000 (and up to 100,000) local descriptors are computed from a dense grid at multiple scales. Changing a single pixel affects multiple overlapping local descriptors, which need to be recomputed prior to a re-evaluation of f. It can be assumed that similar descriptors l and l' (due to the spatially orderless nature of L) will be closely co-located to each other in the image and will be attributed similar relevance values $R^{(2)}_l$ and $R^{(2)}_{l'}$. Thus, replacing one pixel at a time would have the consequence of repeated recomputation of the same subset of L with little effect on the evaluation of f. An obvious solution to this is to reduce the frequency of local descriptor recomputation by chaining perturbations to sets of (neighboring) pixels at a time. This however brings forth the question of (2) how to define a group of pixels and how to perform the perturbation without the introduction of a bias towards a certain object class. *E.g.* a replacement of square image areas with random pixel values might cause an increase of f for non-organic object categories such as "car" in conjunction with SIFT features due to the introduced (on average) gray color and straight edges.

To avoid both above problems we perturb the set of local descriptors $\{l\} = L$ representing the image, instead of pixels. We compute an ordering of local descriptors $\mathcal{O} = (l_i, \ldots, l_{|L|})$ based on their respective relevance values $R^{(2)}_{l_i} = \mathcal{H}(\mathbf{x}, f, l_i)$. This allows for an independent replacement of l_i in L without interfering with other local descriptors. To avoid sampling arbitrary (and nonsensical) descriptors l we use the GMM $\lambda = (\pi_k, \mu_k, \Sigma_k)_{k=1..K}$ as a generator function for replacement descriptors l_λ close to the data manifold. Since the parameters of λ have been fit to a large number of local descriptors from all object categories, high feature diversity among all sampled l_λ can be expected. The perturbation algorithm from Equation

(3.6) is then initialized as

$$x_{\text{MoRF}}^{(0)} = \frac{1}{|L|} \sum_{l \in L} \Psi_\lambda(l) \,, \tag{C.1}$$

the *unnormalized* Fisher vector (see Equation (4.5)). We specify the function g as

$$\begin{aligned} g : \left(x_{\text{MoRF}}^{(k-1)}, l_k\right) &\to x_{\text{MoRF}}^{(k)} \\ x_{\text{MoRF}}^{(k)} &= x_{\text{MoRF}}^{(k-1)} + \frac{1}{|L|}\Psi_\lambda(l_\lambda) - \frac{1}{|L|}\Psi_\lambda(l_k) \,, \end{aligned} \tag{C.2}$$

where $x_{\text{MoRF}}^{(i)}$ is an i times perturbed *unnormalized* Fisher vector. The function g performs a step of perturbation on $x_{\text{MoRF}}^{(k-1)}$ by replacing a descriptor l_k from L in its representation in the Fisher vector feature space with a generated substitute l_λ sampled from λ. To assess the effect of the perturbation we evaluate $f(\Phi(x_{\text{MoRF}}^{(k)}))$ correspondingly (see Equation (4.6)) in Equation (3.7).

When computing the AOPC for an image, we replace the first (wrt $\mathcal{O}$) 10,000 local descriptors l_k in batches of 100, replacing 14.5% of all original descriptors on average. For each image we repeat the process five times in order to reduce the effect of randomness.

Appendix D

Investigating Pretraining and Preprocessing on DNNs for Face Categorization

D.1 Increasing Decomposition Depth for the Fisher Vector SVM Predictor

Chapter 5, Section 5.4.2 has introduced the $\flat$-decomposition rule for LRP in Equation (5.1) as an alternative decomposition variant for layers of Deep Neural Networks. In Chapter 2, Section 2.4.2 the rule hast been applied (heuristically) as a necessity for Bag of Words type models, since in general there is no clear (weighted) relationship between individual pixels and local descriptor dimensions, *e.g.* as it is the case with the quantile-based descriptors used in (Binder et al., 2013). Chapter 4, Section 4.3 follows this heuristic for decomposing local feature relevance values $R_l^{(2)}$ to pixel relevance values $R_p^{(1)}$ for the analyzed Fisher vector model.

For some configurations of Bag of Words model pipelines, which include the Fisher vector model from Chapter 4, a furthered (weighted) decomposition of $R_l^{(2)}$ is possible to obtain relevance maps of higher frequency and finer granularity.

Two requirements must be met for a non-heuristic relevance decomposition of the computation of local descriptors: (1) Relevance scores for each dimension of a local descriptor l must be computable and (2) individual components of the local descriptor l must be relatable to a discrete set of pixel coordinates. For the FV model used in (Chatfield et al., 2011) and Chapter 4, requirement (1) is met fully and (2) partially, as will be explained below.

D.1.1 Computing Relevance Values for Individual Local Feature Dimensions

In order to compute relevance scores for each local feature dimension individually, the forward mapping from each local feature component l_i to each mapping output d needs to be known[1] . (Chatfield et al., 2011) compute mappings

$$\Psi_{\mu_k}(l) = \frac{1}{\sqrt{\pi_k}}\gamma_k(l)\left(\frac{l-\mu_k}{\sigma_k}\right) \tag{D.1}$$

$$\Psi_{\sigma_k}(l) = \frac{1}{\sqrt{2\pi_k}}\gamma_k(l)\left(\frac{(l-\mu_k)^2}{\sigma_k^2}-1\right) \tag{D.2}$$

from l to all K components of a GMM $\lambda = \{(\pi_k, \mu_k, \Sigma_k)\}_{1..K}$ wrt to its 1st and 2nd moments, with $\{l, \Psi_{\mu_k}(l), \Psi_{\sigma_k}(l)\} \in \mathbb{R}^D$.

Further, the convariance matrices of the trained GMM have been constrained to be diagonal (*e.g.* $\forall k : \Sigma_k = diag(\sigma_k)$). Therefore it is known that each l_i corresponds to exactly

[1] Either explicity, as it is the case here with the Fisher vector model, or via a distance measure decomposable into dimensional contributions such as the squared euclidean distance often used for visual word assignments in the computation of (soft) Bag of Words mappings. See Section 2.4.3.

one dimension in the mapping output space of $\Psi_{\mu_k}(l)$ and $\Psi_{\sigma_k}(l)$ for all k. Suppose a function $\delta(i, \Psi_{\{\mu,\sigma\}_k})$, which computes for a local feature dimension i and mapping type (*i.e.* Ψ_{μ_k} or Ψ_{σ_k}) the output dimension d of the FV representation. We then can compute relevance scores (without detour) for each l_i as

$$R^{(2)}_{l_i} = \sum_{k=1}^{K} \left(\frac{\Psi_{\mu_k}(l)_i}{\sum_{l' \in L} \Psi_{\mu_k}(l')_i} R^{(3)}_{\delta(i,\Psi_{\mu_k})} + \frac{\Psi_{\sigma_k}(l)_i}{\sum_{l' \in L} \Psi_{\sigma_k}(l')_i} R^{(3)}_{\delta(i,\Psi_{\sigma_k})} \right) . \tag{D.3}$$

Note that in practice we do still apply the ϵ-stabilized decomposition variant from Equation (2.21).

Prior to mapping the local descriptors into the FV space, the reference model from (Chatfield et al., 2011) performs dimensionality reduction on the initial 128-dimensional SIFT (Lowe, 2004) features onto an 80-dimensional subspace via PCA (Pearson, 1901) during training (with mapping components being computed once over the whole training set). Since a projection with PCA is nothing more than a linear mapping, corresponding relevance decomposition can be handled as such (*i.e.* with Equation (2.31)).

D.1.2 From Relevance Scores for Local Feature Dimensions to Individual Pixels

Many local feature types aggregate information extracted from an image area, such that the relation between feature dimension and pixel coordinate is lost in the process, *e.g.* due quantiles computed over larger sets of pixels. However, information about local feature geometry can be used whenever possible in order to appropriately aggregate $R^{(2)}_{l_i}$ (over several i) and assign relevance scores to pixels at a higher resolution compared to the heuristic $\flat$-decomposition approach. The FV predictor from (Chatfield et al., 2011) and Chapter 4 uses SIFT descriptors at different sizes with (4×4) spatial bins each, covering roughly square groups of pixels. Over each spatial bin, a histogram of gradient magnitudes in 8 directions is computed.

We use the known feature geometry (Vedaldi et al., 2010) to distribute the sum-aggregated relevances corresponding to each spatial bin evenly over the covered pixels, thus increasing the resolution of the decomposed relevance values from each SIFT feature 16-fold.

D.1.3 Examples for Relevance Maps of different Decomposition Depth for Fisher Vector and DNN Models

Example relevance maps for the DNN model used in Chapter 4 in comparison to relevance maps for the Fisher vector model are shown in Figure D.1. For the Fisher vector predictor a further decomposition step according to Equation D.3 has been added, while for the DNN model, the $\flat$-decomposition rule (Equations (2.27) and (5.1)) has been applied to the lowest convolution layer.

The increased relevance map resolution for the Fisher vector classifier provides insights about structures important for prediction at a smaller scale. In the example for class "chair" in Figure D.1 the high resolution relevance map demonstrates that the FV classifier dominantly uses the object structure itself for classification, which was more difficult to perceive from the lower resolution relevance maps. Both models seem to follow similar higher level strategies for most object classes, *e.g.* predicting cars based on the lower chassis, predicting people based on face and clothing (texture), predicting bicycles based on the tires, and predicting dogs based on facial features.

Both classifiers seem to prefer the use of edges with high contrast within the images (*e.g.* cutlery in samples of class "diningtable") for prediction, as learned by the DNN during training and via by design via the choice of local features (SIFT) for the Fisher vector model.

The more deep and complex DNN classifier seems to be in general more adept at abstracting object appearances and is therefore less prone to misleading (and isolated bits of) information compared to the Fisher vector model. At hand of the example showing a dog we can observe the Fisher vector model's decision based on the dog's eyes, but also some small black spots in the snowy background, resembling eyes and nose. This can be explained

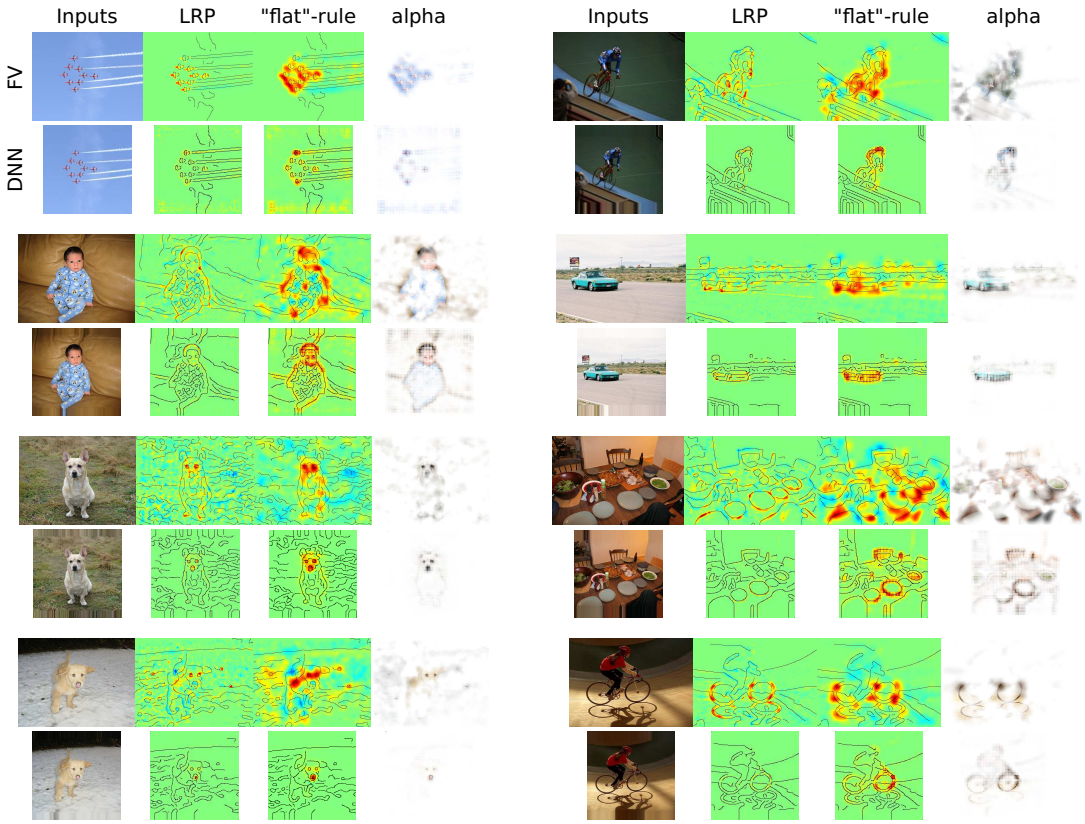

FIGURE D.1: **Inputs for DNN and Fisher vector models and relevance maps with $\flat$-rule decomposition applied at different layers.** Each group of images shows in the top row inputs and relevance maps corresponding to the Fisher vector (FV) model and in the bottom row data corresponding to the DNN. The first of the of the four columns (*Inputs*) in reach row shows the model input. The second column (*LRP*) shows relevance maps computed as far as possible to the pixel level. The third column ($\flat$-*rule*) shows relevance maps computed with the $\flat$-rule decomposition in the bottom layer of computation. The fourth column (*alpha*) shows the input, using the the positive relevance values of the relevance maps in the third column as the image's alpha channel, highlighting the important features of the image. Images and model predictions correspond to the Pascal VOC dataset.

in the model's feature extraction paradigm being based on a set of spatially orderless SIFT descriptors: While some descriptors are covering the image areas showing the black spots in the background are picking up information matching the criteria of "a dog's facial features" the do not collect any further information about spatial composition, other than the convolution layers of the DNN.

D.2 The Effects of Preprocessing and Dataset Composition to Age Categorization Tasks

Chapter 5 names the choice of face image alignment as one of the main factors influencing model performances and Figure 5.5 points out some of the distortions occuring in the Adience dataset (Eidinger et al., 2014) under in-plane alignment. The confusion matrices for the CaffeNet, GoogleNet and VGG-16 models in Figure D.2 show that (almost[2]) all models and age classes do not benefit from in-plain aligmnent as a preprocessing choice. Most notably samples representing young children aged (4-6) and (8-13) benefit from (the addition

[2]The only exception is the age group (48-53) with the GoogleNet and the VGG-16 models.

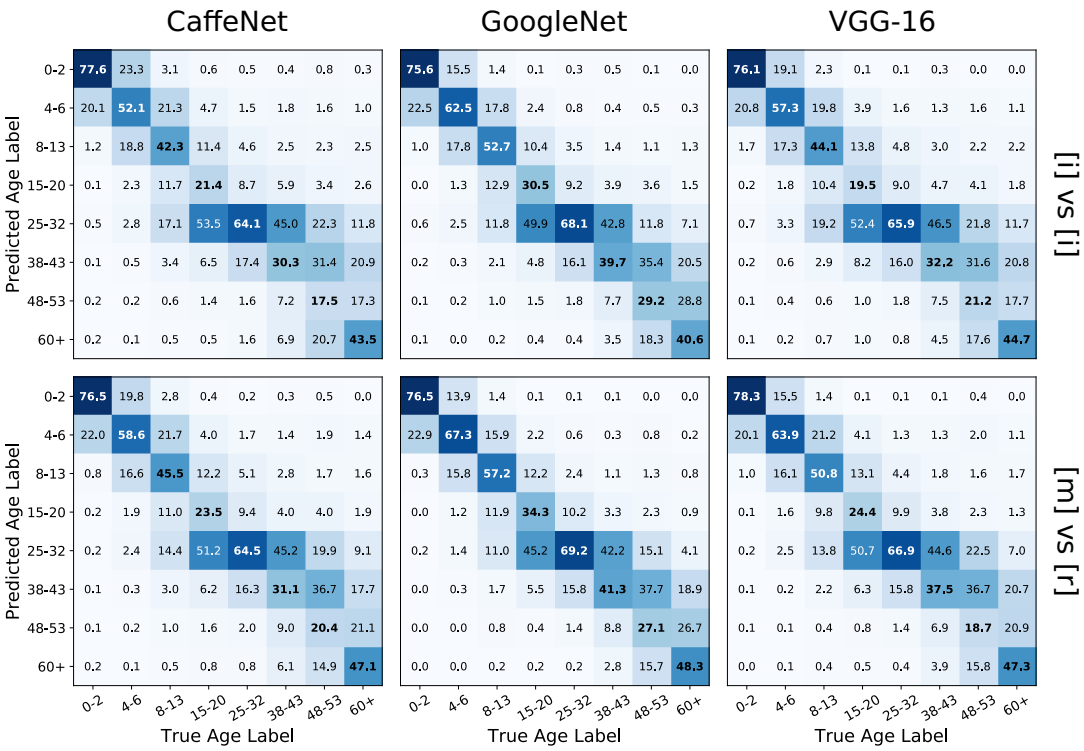

FIGURE D.2: **Age prediction confusion matrices for the CaffeNet, GoogleNet and VGG-16 model comparing predictions under different preprocessing settings.** All models have been pre-trained on ImageNet only. Values are in percent. *Top row:* Confusion matrices for training and testing with samples under [i]n-plane alignment. *Bottom row:* Confusion matrices for training with data from [m]ixed alignments and prediction on [r]otation aligned samples.

of) rotation alignment, resulting in up to +6.6% true positive predictions due to reduced confusion with adjacend age groups, but also the group of (25-32) year olds.

Least affected by the change in preprocessing is the class representing ages (25-32), which also provides by far the highest amount of training samples in the data set. Such a population imbalance is not an issue unique to the Adience dataset, but also affects the orders of magnitude larger IMDB-WIKI (Rothe et al., 2016) dataset. Figure D.3 provides an overview over the age label distributions in both datasets. In all three shown data sources, the population of 20 to 40 year olds is represented the most, due to (voluntary) exposition of humans in that age group in media and social media. The confusion matrices in Figure D.2 show that all models – regardless of preprocessing choice – tend to dominantly predict samples from classes (15-20) and (38-43) as the adjacent and overpopulated class (25-32) due to the existing class imbalance.

We can use LRP to analyze which facial features are responsible for the decision for or against certain age classes. Figure D.4 presents relevance maps for four face images – two male and two female – as computed for the VGG-16 model pretrained on (only) the ImageNet data. From the shown relevance map we hypothesize that the model has learned that smiling is an indicator for ages 15 to 43 only. So is *e.g.* the woman in the second row predicted as a member of the age group (60-100) due to the majority of her facial appearance, with the exception of her smile and the visible teeth, which contradicts old age according to the model. The bottom two rows of Figure D.4 indicate that beards, glasses and baldness are expected to appear as aging progresses, while t-shirt collars and colorful backgrounds of high contrast are features connected to more juvenile age groups.

While the latter two rows of Figure D.4 show intutively meaningful explanations of the model's decisions regarding age group related features, rows one and two indicate a potential bias in the model's reasoning: Apparently smiling is reserved as a describing feature

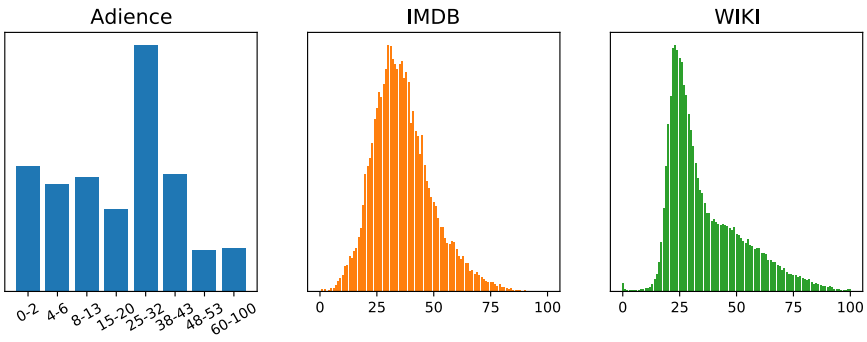

FIGURE D.3: **Age label distributions over the populations of datasets encoding age recognition tasks from face images.** *From left to right:* Label distributions of the eight age groups of the Adience dataset (Eidinger et al., 2014) and the 101 age classes IMDB and WIKI(pedia) parts of the IMDB-WIKI dataset (Rothe et al., 2016). In all three dataset, the majority of the polulation is made up of faces of (approximately) 25 to 35 years of age.

for ages 15 to 43, which may be explained with typical photography settings of and media affine (which may also be a reason for the high sample count representing age group (25-32)) young people enjoying festivities. We conclude that the Adience benchmark dataset could be improved by including more happy children and older people to prevent models falling for the smiles as a learned bias for predicting ages around the early 30s.

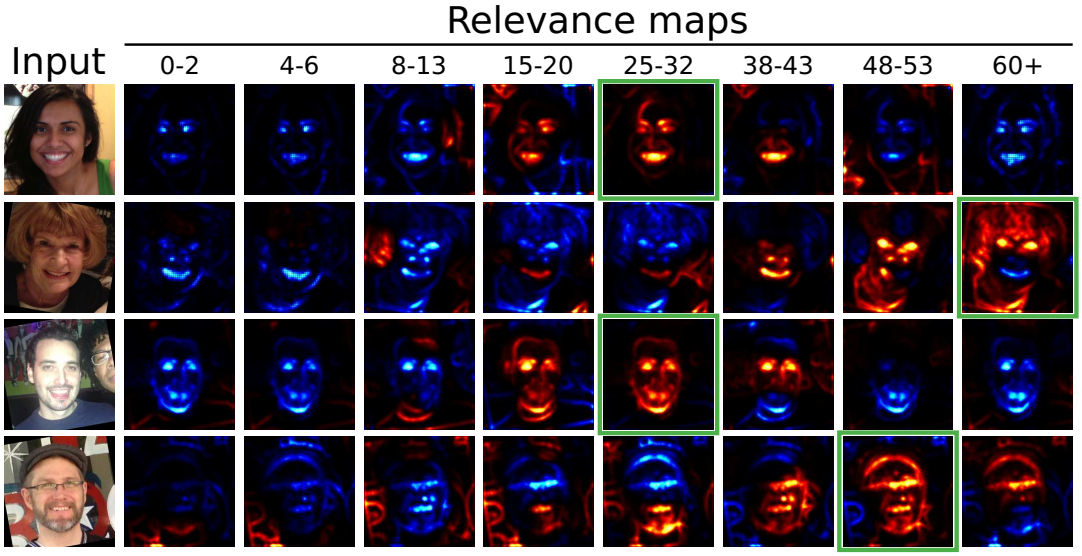

FIGURE D.4: **Relevance maps for a range of inputs from the Adience dataset and a VGG16-model pretrained on ImageNet.** For the inputs to the left, relevance maps are computed for all eight age classes of the Adience face recognition benchmark dataset. The dominantly predicted class for each input is indicated by a green box around the class relevance map.

Appendix E

Ensemble Relevance Map Analysis of Classifier Behaviour

We formulate the methodology of SpRAy algorithmically with Algorithm 3 and present some additional results obtained with SpRAy with a low pass filtering component added to the preprocessing via Gaussian smoothing. Figures E.1 and E.2 show eigenvalues, cluster assignments and embeddings for class "horse" and Figures E.3 and E.4 show eigenvalues, cluster assignments and embeddings for class "aeroplane".

Algorithm 3: Spectral Relevance Analysis

Data: Input samples $X = \{\boldsymbol{x}\}$, a model f operating on X.
Result: eigenvalues $\Lambda = \{\lambda\}$, cluster labels $Y = \{y\}$, embeddings $Z = \{\boldsymbol{z}\}$

```
/* compute and collect relevance maps for x ∈ X                    */
R = {};
for x ∈ X do
    R_x = LRP(f, x);
    R.add(R_x);
end
/* preprocess relevance maps in R                                  */
for R_x ∈ R do
    R_x ← preprocess(R_x);
end
/* compute weighted affinity matrix and laplacian                  */
W = weighted_affinity(R);
L = laplacian(W);
/* compute eigenvalues Λ cluster labels Y and embedding
   coordinates Z for further visualization and analysis.           */
Λ, Y = spectral(L);
D = affinity2distance(W);
Z = tsne(D);
```

The analysis over finely structured relevance maps obtained for the DNN models does in general not benefit from the low pass filtering step, since the high frequency relevance information vanishes and the relationship between samples becomes almost one-dimensional via the average relevance score of each sample. To relevance maps obtained for the FV model, the additional preprocessing step may be beneficial. SpRAy is able to identify the watermark-based prediction artefact for class "horse", yet seems unable to differentiate between affected images in landscape format and portrait format. This can be seen in the two eigenvalues close to zero, before the first larger eigengap in Figure E.1, compared to the four eigenvalues before the first large eigengap without the interpolation step from Figure 6.3 and the corresponding cluster label assignments in Figure E.2. For class "aeroplane", the low pass preprocessing step reduces the negative relevance feedback attributed to the smaller dimensioned aeroplanes in the images for the FV model. Instead, SpRAy highlights the tiling structure in the attributed relevance values focussing on the (filtered) low frequency content within the relevance maps. It becomes apparent that some of the samples from test set of

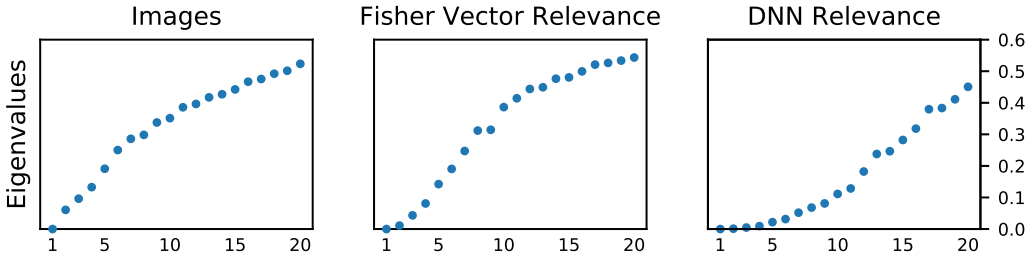

FIGURE E.1: **The first 20 eigenvalues for class "horse" for images, relevance maps for the FV predictor and the DNN respectively, after the inclusion of a low pass filtering step into the preprocessing.**

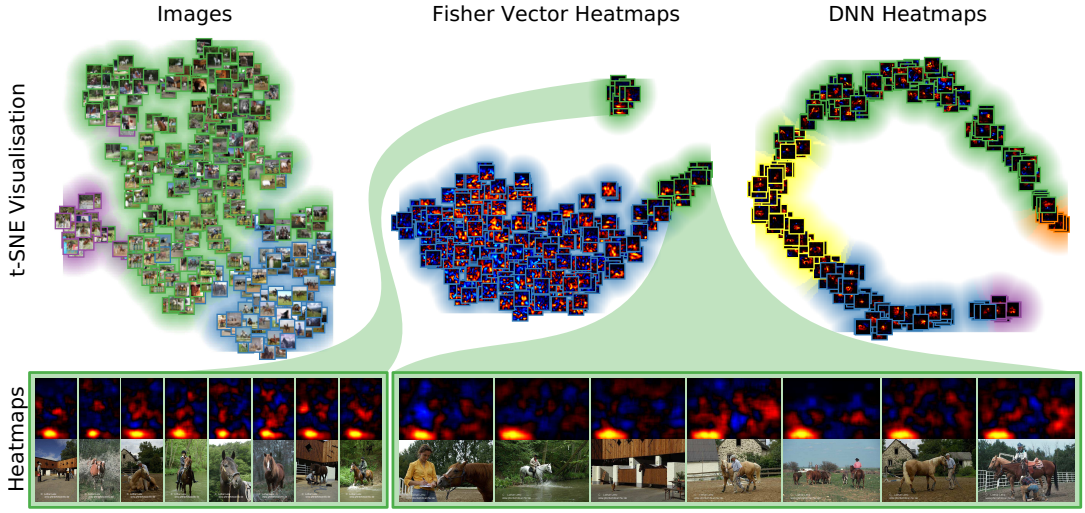

FIGURE E.2: **Cluster label assignments for class "horse" via SC computed from input images (left), FV relevance maps (middle) and DNN relevance maps (right).**

class "aeroplane" clearly are recognized from the image background. Further analyses have verified the spatial pyramid mapping scheme of the FV model (partially) as the cause of this particular prediction behaviour.

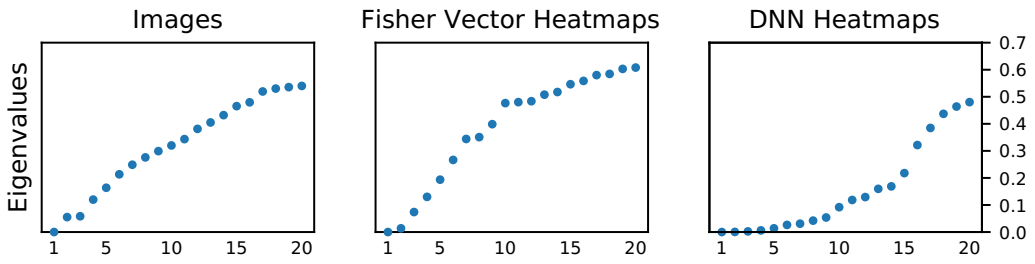

FIGURE E.3: **The first 20 eigenvalues for class "aeroplane" for images, relevance maps for the FV predictor and the DNN respectively, after the inclusion of a low pass filtering step into the preprocessing.**

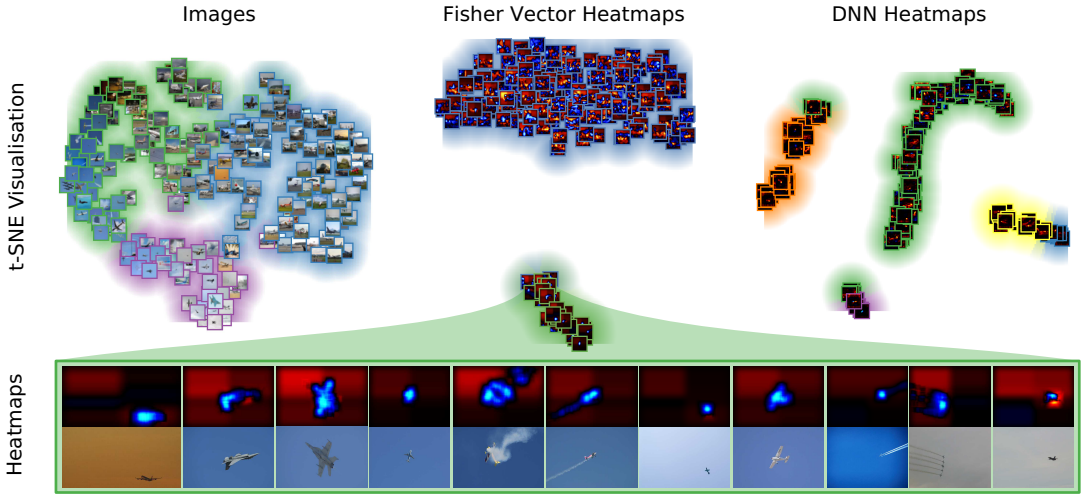

FIGURE E.4: **Cluster label assignments for class "aeroplane" via SC computed from input images (left), FV relevance maps (middle) and DNN relevance maps (right).**

Bibliography

Alber, M., S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, and P.-J. Kindermans (2018a). "How to iNNvestigate Neural Networks' predictions!" In: *Machine Learning Open Source Software (MLOSS): Sustainable Communities, NIPS Workshop*.

— (2018b). "iNNvestigate Neural Networks!" In: *CoRR* abs/1808.04260. arXiv: 1808.04260.

Amarasinghe, K., K. Kenney, and M. Manic (2018). "Toward Explainable Deep Neural Network Based Anomaly Detection". In: *11th International Conference on Human System Interaction, HSI 2018, Gdansk, Poland, July 4-6, 2018*, pp. 311–317.

Ancona, M., E. Ceolini, C. Öztireli, and M. Gross (2018). "Towards Better Understanding of Gradient-based Attribution Methods for Deep Neural Networks". In: *International Conference of Learning Representations (ICLR)*.

Anders, C., G. Montavon, W. Samek, and K.-R. Müller (2018). "Understanding Patch-Based Learning by Explaining Predictions". In: *CoRR* abs/1806.06926. arXiv: 1806.06926.

Arbabzadah, F., G. Montavon, K.-R. Müller, and W. Samek (2016). "Identifying Individual Facial Expressions by Deconstructing a Neural Network". In: *Pattern Recognition - 38th German Conference, GCPR 2016, Hannover, Germany, September 12-15, 2016, Proceedings*, pp. 344–354.

Arp, D., M. Spreitzenbarth, M. Hübner, H. Gascon, K. Rieck, and C. Siemens (2014). "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket". In: *NDSS*.

Arras, L., F. Horn, G. Montavon, K.-R. Müller, and W. Samek (2016). "Explaining Predictions of Non-Linear Classifiers in NLP". In: *Proceedings of the 1st Workshop on Representation Learning for NLP, Rep4NLP@ACL 2016, Berlin, Germany, August 11, 2016*, pp. 1–7.

— (2017a). ""What is Relevant in a Text Document?": An Interpretable Machine Learning Approach". In: *PloS ONE* 12.8, e0181142.

Arras, L., G. Montavon, K.-R. Müller, and W. Samek (2017b). "Explaining Recurrent Neural Network Predictions in Sentiment Analysis". In: *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pp. 159–168.

Bach, F. R., G. R. G. Lanckriet, and M. I. Jordan (2004). "Multiple Kernel Learning, Conic Duality, and the SMO Algorithm". In: *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*.

Bach, S. (2013). *On Pixel-wise Predictions from Image-wise Bag of Words Classification*. Berlin Institute of Technology. Master's Thesis.

Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek (2015). "On Pixel-wise Explanations for Non-Linear Classifier Decisions by Layer-wise Relevance Propagation". In: *PLoS ONE* 10.7, e0130140.

Bach, S., A. Binder, K.-R. Müller, and W. Samek (2016). "Controlling Explanatory Heatmap Resolution and Semantics via Decomposition Depth". In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pp. 2271–2275.

Baehrens, D., T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller (2010). "How to Explain Individual Classification Decisions". In: *Journal of Machine Learning Research* 11.Jun, pp. 1803–1831.

Baluja, S. and H. A. Rowley (2007). "Boosting Sex Identification Performance". In: *International Journal of Computer Vision* 71.1, pp. 111–119.

Becker, S., M. Ackermann, S. Lapuschkin, K.-R. Müller, and W. Samek (2018). "Interpreting and Explaining Deep Neural Networks for Classification of Audio Signals". In: *CoRR* abs/1807.03418. arXiv: 1807.03418.

Bharadhwaj, H. (2018). "Layer-wise Relevance Propagation for Explainable Recommendations". In: *Proceedings of the ACM SIGIR'18 Workshop on ExplainAble Recommendation and Search (EARS)*.

Binder, A., S. Bach, G. Montavon, K.-R. Müller, and W. Samek (2016a). "Layer-wise Relevance Propagation for Deep Neural Network Architectures". In: *Information Science and Applications (ICISA) 2016*. Ed. by K. J. Kim and N. Joukov. Vol. 376. Lecture Notes in Electrical Engineering. Singapore: Springer Singapore, pp. 913–922. ISBN: 978-981-10-0557-2.

Binder, A., M. Bockmayr, M. Hägele, S. Wienert, D. Heim, H. Katharina, A. Stenzinger, L. Parlow, J. Budczies, B. Goeppert, D. Treue, M. Kotani, M. Ishii, M. Dietel, A. Hocke, C. Denkert, K.-R. Müller, and F. Klauschen (2018). "Towards Computational Fluorescence Microscopy: Machine Learning-based Integrated Prediction of Morphological and Molecular Tumor Profiles". In: *CoRR* abs/1805.11178. arXiv: 1805.11178.

Binder, A., G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek (2016b). "Layer-wise Relevance Propagation for Neural Networks with Local Renormalization Layers". In: vol. 9887. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 63–71.

Binder, A., K.-R. Müller, and M. Kawanabe (2012a). "On Taxonomies for Multi-class Image Categorization". In: *International Journal of Computer Vision* 99.3, pp. 281–301.

Binder, A., S. Nakajima, M. Kloft, C. Müller, W. Samek, U. Brefeld, K.-R. Müller, and M. Kawanabe (2012b). "Insights from Classifying Visual Concepts with Multiple Kernel Learning". In: *PloS ONE* 7.8, e38897.

Binder, A., W. Samek, G. Montavon, S. Bach, and K.-R. Müller (2016c). "Analyzing and Validating Neural Networks Predictions". In: *Proceedings of the ICML'16 Workshop on Visualization for Deep Learning*, pp. 1–4.

Binder, A., W. Samek, K.-R. Müller, and M. Kawanabe (2013). "Enhanced Representation and Multi-task Learning for Image Annotation". In: *Computer Vision and Image Understanding* 117.5, pp. 466–478.

Blankertz, B., S. Lemm, M. S. Treder, S. Haufe, and K. Müller (2011). "Single-trial Analysis and Classification of ERP Components - A Tutorial". In: *NeuroImage* 56.2, pp. 814–825.

Blankertz, B., R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Müller (2008). "Optimizing Spatial Filters for Robust EEG Single-trial Analysis". In: *IEEE Signal processing magazine* 25.1, pp. 41–56.

Blaschko, M. B., W. Zaremba, and A. Gretton (2013). "Taxonomic Prediction with Tree-Structured Covariances". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part II*, pp. 304–319.

Bojarski, M., A. Choromanska, K. Choromanski, B. Firner, L. J. Ackel, U. Muller, P. Yeres, and K. Zieba (2018). "VisualBackProp: Efficient Visualization of CNNs for Autonomous Driving". In: *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pp. 1–8.

Bosch, A., A. Zisserman, and X. Munoz (2007). "Representing Shape with a Spatial Pyramid Kernel". In: *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*. ACM, pp. 401–408.

Boser, B. E., I. Guyon, and V. Vapnik (1992). "A Training Algorithm for Optimal Margin Classifiers". In: *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT 1992, Pittsburgh, PA, USA, July 27-29, 1992*. Pp. 144–152.

Braun, M. L., J. M. Buhmann, and K.-R. Müller (2008). "On Relevant Dimensions in Kernel Feature Spaces". In: *Journal of Machine Learning Research* 9.Aug, pp. 1875–1908.

Burrell, J. (2016). "How the Machine "Thinks": Understanding Opacity in Machine Learning Algorithms". In: *Big Data & Society* 3.1, p. 2053951715622512.

Calvillo, J. and M. Crocker (2018). "Language Production Dynamics with Recurrent Neural Networks". In: *Proceedings of the Eight Workshop on Cognitive Aspects of Computational Language Learning and Processing*, pp. 17–26.

Cao, L., J. Luo, F. Liang, and T. S. Huang (2009). "Heterogeneous Feature Machines for Visual Recognition". In: *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pp. 1095–1102.

Caruana, R., Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad (2015). "Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pp. 1721–1730.

Chang, F., A. T. Tran, T. Hassner, I. Masi, R. Nevatia, and G. G. Medioni (2017). "FacePoseNet: Making a Case for Landmark-Free Face Alignment". In: *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017*, pp. 1599–1608.

Chang, W.-C., C.-P. Lee, and C.-J. Lin (2013). "A Revisit to Support Vector Data Description". In: *Dept. Comput. Sci., Nat. Taiwan Univ., Taipei, Taiwan, Tech. Rep.*

Chatfield, K., V. S. Lempitsky, A. Vedaldi, and A. Zisserman (2011). "The Devil is in the Details: An Evaluation of Recent Feature Encoding Methods". In: *British Machine Vision Conference, BMVC 2011, Dundee, UK, August 29 - September 2, 2011. Proceedings*, pp. 1–12.

Chatfield, K., K. Simonyan, A. Vedaldi, and A. Zisserman (2014). "Return of the Devil in the Details: Delving Deep into Convolutional Nets". In: *British Machine Vision Conference, BMVC 2014, Nottingham, UK, September 1-5, 2014*.

Chong, P., Y. X. M. Tan, J. Guarnizo, Y. Elovici, and A. Binder (2018). "Mouse Authentication Without the Temporal Aspect – What Does a 2D-CNN Learn?" In: *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, pp. 15–21.

Clevert, D.-A., T. Unterthiner, and S. Hochreiter (2016). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". In: *International Conference of Learning Representations (ICLR)*.

Cortes, C. and V. Vapnik (1995). "Support-vector Networks". In: *Machine learning* 20.3, pp. 273–297.

Cosman, P. C., K. L. Oehler, E. A. Riskin, and R. M. Gray (1993). "Using Vector Quantization for Image Processing". In: *Proceedings of the IEEE* 81.9, pp. 1326–1341.

Dehghan, A., E. G. Ortiz, G. Shu, and S. Z. Masood (2017). "DAGER: Deep Age, Gender and Emotion Recognition Using Convolutional Neural Network". In: *CoRR* abs/1702.04280. arXiv: 1702.04280.

Ding, Y., Y. Liu, H. Luan, and M. Sun (2017). "Visualizing and Understanding Neural Machine Translation". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pp. 1150–1159.

Dosovitskiy, A. and T. Brox (2016). "Inverting Visual Representations with Convolutional Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 4829–4837.

Dreyfus, H. L. and S. E. Dreyfus (1992). "What Artificial Experts Can and Cannot Do". In: *AI & society* 6.1, pp. 18–26.

Eidinger, E., R. Enbar, and T. Hassner (2014). "Age and Gender Estimation of Unfiltered Faces". In: *IEEE Trans. Information Forensics and Security* 9.12, pp. 2170–2179.

Erhan, D., A. Courville, and Y. Bengio (2010). "Understanding Representations Learned in Deep Architectures". In: *Department d'Informatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep* 1355.

Escalera, S., J. Fabian, P. Pardo, X. Baró, J. Gonzàlez, H. J. Escalante, D. Misevic, U. Steiner, and I. Guyon (2015). "ChaLearn Looking at People 2015: Apparent Age and Cultural Event Recognition Datasets and Results". In: *2015 IEEE International Conference on Computer Vision Workshop, ICCV Workshops 2015, Santiago, Chile, December 7-13, 2015*, pp. 243–251.

Everingham, M., L. Van Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman (2009). "The PASCAL Visual Object Classes Challenge 2009 (VOC2009)". In: *Summary presentation at the 2009 PASCAL VOC workshop*. Vol. 10.

— (2010). "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88.2, pp. 303–338.

Fan, L., S. Zhao, and S. Ermon (2017). "Adversarial Localization Network". In: *Learning with Limited Labeled Data: Weak Supervision and Beyond, NIPS Workshop*.

Firoiu, V., W. F. Whitney, and J. B. Tenenbaum (2017). "Beating the World's Best at Super Smash Bros. with Deep Reinforcement Learning". In: *CoRR* abs/1702.06230. arXiv: 1702.06230.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems". In: *Annals of human genetics* 7.2, pp. 179–188.

Fong, R. C. and A. Vedaldi (2017). "Interpretable Explanations of Black Boxes by Meaningful Perturbation". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 3449–3457.

Gallagher, A. C. and T. Chen (2008). "Clothing Cosegmentation for Recognizing People". In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*.

— (2009). "Understanding Images of Groups of People". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 256–263.

Gao, F. and H. Ai (2009). "Face Age Classification on Consumer Images with Gabor Feature and Fuzzy LDA Method". In: *Advances in Biometrics, Third International Conference, ICB 2009, Alghero, Italy, June 2-5, 2009. Proceedings*, pp. 132–141.

Gemert, J. C. van, J. Geusebroek, C. J. Veenman, and A. W. M. Smeulders (2008). "Kernel Codebooks for Scene Categorization". In: *Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III*, pp. 696–709.

Gemert, J. C. van, C. J. Veenman, A. W. M. Smeulders, and J. Geusebroek (2010). "Visual Word Ambiguity". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 32.7, pp. 1271–1283.

Girshick, R. B., J. Donahue, T. Darrell, and J. Malik (2014). "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*, pp. 580–587.

Goodfellow, I. J., J. Shlens, and C. Szegedy (2014). "Explaining and Harnessing Adversarial Examples". In: *CoRR* abs/1412.6572. arXiv: 1412.6572.

Goodman, B. and S. R. Flaxman (2017). "European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation"". In: *AI Magazine* 38.3, pp. 50–57.

Gotsopoulos, A., H. Saarimäki, E. Glerean, I. P. Jääskeläinen, M. Sams, L. Nummenmaa, and J. Lampinen (2018). "Reproducibility of Importance Extraction Methods in Neural Network based fMRI Classification". In: *NeuroImage* 181, pp. 44–54.

Grave, E., T. Mikolov, A. Joulin, and P. Bojanowski (2017). "Bag of Tricks for Efficient Text Classification". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pp. 427–431.

Graves, A., G. Wayne, M. Reynolds, T. Harley, I. Danihelka, A. Grabska-Barwińska, S. G. Colmenarejo, E. Grefenstette, T. Ramalho, J. Agapiou, et al. (2016). "Hybrid Computing using a Neural Network with Dynamic External Memory". In: *Nature* 538.7626, pp. 471–476.

Guodong, G., G. Mu, Y. Fu, and T. S. Dyer Charles R.and Huang (2009). "A Study on Automatic Age Estimation using a Large Database". In: *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pp. 1986–1991.

Guyon, I. and A. Elisseeff (2003). "An Introduction to Variable and Feature Selection". In: *Journal of machine learning research* 3.Mar, pp. 1157–1182.

Hansen, K., D. Baehrens, T. Schroeter, M. Rupp, and K.-R. Müller (2011). "Visual Interpretation of Kernel-based prediction models". In: *Molecular Informatics* 30.9, pp. 817–826.

Hassner, T., S. Harel, E. Paz, and R. Enbar (2015). "Effective Face Frontalization in Unconstrained Images". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pp. 4295–4304.

Haufe, S., F. C. Meinecke, K. Görgen, S. Dähne, J.-D. Haynes, B. Blankertz, and F. Bießmann (2014). "On the Interpretation of Weight Vectors of Linear Models in Multivariate Neuroimaging". In: *NeuroImage* 87, pp. 96–110.

He, K., X. Zhang, S. Ren, and J. Sun (2016). "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Heeger, D. J., E. P. Simoncelli, and J. A. Movshon (1996). "Computational Models of Cortical Visual Processing". In: *Proceedings of the National Academy of Sciences* 93.2, pp. 623–627.

Hochreiter, S. and J. Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780.

Hochuli, J., A. Helbling, T. Skaist, M. Ragoza, and D. R. Koes (2018). "Visualizing Convolutional Neural Network Protein-ligand Scoring". In: *Journal of Molecular Graphics and Modelling*.

Horn, F., L. Arras, G. Montavon, K.-R. Müller, and W. Samek (2017). "Exploring Text Datasets by Visualizing Relevant Words". In: *CoRR* abs/1707.05261. arXiv: 1707.05261.

Horst, F., M. Mildner, and W. I. Schöllhorn (2017). "One-year Persistence of Individual Gait Patterns Identified in a Follow-up Study – A Call for Individualised Diagnose and Therapy". In: *Gait & posture* 58, pp. 476–480.

Horst, F., S. Lapuschkin, W. Samek, K.-R. Müller, and W. I. Schöllhorn (2018). "Explaining the Unique Nature of Individual Gait Patterns With Deep Learning". In: *Scientific Reports*. In Revision.

Huang, G., Z. Liu, L. van der Maaten, and K. Q. Weinberger (2017). "Densely Connected Convolutional Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2261–2269.

Huang, G. B., M. Mattar, T. Berg, and E. Learned-Miller (2008). "Labeled Faces in the Wild: A Database for Studying Dace Recognition in Unconstrained Environments". In: *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*.

Hunter, J. D. (2007). "Matplotlib: A 2D Graphics Environment". In: *Computing in Science and Engineering* 9.3, pp. 90–95.

Hwang, S. J., K. Grauman, and F. Sha (2012). "Semantic Kernel Forests from Multiple Taxonomies". In: *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* Pp. 1727–1735.

Ioffe, S. and C. Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *International Conference on Machine Learning*, pp. 448–456.

Itti, L. and C. Koch (2000). "A Saliency-based Search Mechanism for Overt and Covert Shifts of Visual Attention". In: *Vision research* 40.10-12, pp. 1489–1506.

— (2001). "Computational Modelling of Visual Attention". In: *Nature reviews neuroscience* 2.3, p. 194.

Jia, Y., E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell (2014). "Caffe: Convolutional Architecture for Fast Feature Embedding". In: pp. 675–678.

Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features". In: *Machine Learning: ECML-98, 10th European Conference on Machine Learning, Chemnitz, Germany, April 21-23, 1998, Proceedings*, pp. 137–142.

Karpathy, A. (2014). *What I Learned from Competing Against a ConvNet on ImageNet*. https://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet.

Karpathy, A., A. Joulin, and F. Li (2014a). "Deep Fragment Embeddings for Bidirectional Image Sentence Mapping". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 1889–1897.

Karpathy, A., G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. Li (2014b). "Large-Scale Video Classification with Convolutional Neural Networks". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pp. 1725–1732.

Kauffmann, J., K.-R. Müller, and G. Montavon (2018). "Towards Explaining Anomalies: A Deep Taylor Decomposition of One-Class Models". In: *CoRR* abs/1805.06230. arXiv: 1805.06230.

Kauppi, T., V. Kalesnykiene, J.-K. Kamarainen, L. Lensu, I. Sorri, A. Raninen, R. Voutilainen, H. Uusitalo, H. Kälviäinen, and J. Pietilä (2007). "The DIARETDB1 Diabetic Retinopathy Database and Evaluation Protocol". In: *Proceedings of the British Machine Vision Conference 2007, University of Warwick, UK, September 10-13, 2007*, pp. 1–10.

Kelley, D. R., Y. Reshef, M. Bileschi, D. Belanger, C. Y. McLean, and J. Snoek (2018). "Sequential Regulatory Activity Prediction Across Chromosomes With Convolutional Neural Networks." In: *Genome research*, gr–227819.

Kindermans, P.-J., S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, and B. Kim (2017a). "The (Un)reliability of Saliency Methods". In: *CoRR* abs/1711.00867. arXiv: 1711.00867.

Kindermans, P.-J., K. T. Schütt, M. Alber, K.-R. Müller, and S. Dähne (2017b). "Learning how to Explain Neural Networks: PatternNet and PatternAttribution". In: *CoRR* abs/1705.05598v2. arXiv: 1705.05598.

Kindermans, P.-J., K. T. Schütt, K.-R. Müller, and S. Dähne (2016). "Investigating the Influence of Noise and Distractors on the Interpretation of Neural Networks". In: *CoRR* abs/1611.07270. arXiv: 1611.07270.

Klauschen, F., K.-R. Müller, A. Binder, M. Bockmayr, M. Hägele, P. Seegerer, S. Wienert, G. Pruneri, S. de Maria, S. Badve, et al. (2018). "Scoring of Tumor-infiltrating Lymphocytes:

From Visual Estimation to Machine Learning". In: *Seminars in Cancer Biology*. Vol. 52.2. Elsevier, pp. 151–157.

Kloft, M., U. Brefeld, S. Sonnenburg, and A. Zien (2011). "l_p-Norm Multiple Kernel Learning". In: *Journal of Machine Learning Research* 12, pp. 953–997.

Kloft, M., B. Ulf, S. Sonnenburg, P. Laskov, K.-R. Müller, and A. Zien (2009). "Efficient and Accurate ℓ_p-Norm Multiple Kernel Learning". In: *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.* Pp. 997–1005.

Kohlbrenner, M. H. (2017). *On the Stability of Neural Network Explanations*. Berlin Institute of Technology. Bachelor's Thesis.

Koutník, J., G. Cuccu, J. Schmidhuber, and F. J. Gomez (2013). "Evolving Large-scale Neural Networks for Vision-based Reinforcement learning". In: *Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6-10, 2013*, pp. 1061–1068.

Krizhevsky, A. (2009). *Learning Multiple Layers of Features from Tiny Images*. University of Toronto. Master's Thesis.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImagenNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1097–1105.

Kuehne, H., H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre (2011). "HMDB: A Large Video Database for Human Motion Recognition". In: *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pp. 2556–2563.

Kwon, Y. H. and N. da Vitoria Lobo (1994). "Age Classification from Facial Images". In: *Conference on Computer Vision and Pattern Recognition, CVPR 1994, 21-23 June, 1994, Seattle, WA, USA*, pp. 762–767.

Lanckriet, G. R. G., N. Cristianini, P. L. Bartlett, L. El Ghaoui, and M. I. Jordan (2004). "Learning the Kernel Matrix with Semidefinite Programming". In: *Journal of Machine Learning Research* 5, pp. 27–72.

Lapuschkin, S., A. Binder, G. Montavon, K.-R. Müller, and W. Samek (2016a). "Analyzing Classifiers: Fisher Vectors and Deep Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2912–2920.

— (2016b). "The Layer-wise Relevance Propagation Toolbox for Artificial Neural Networks". In: *Journal of Machine Learning Research* 17.114, pp. 1–5.

Lapuschkin, S., A. Binder, K.-R. Müller, and W. Samek (2017). "Understanding and Comparing Deep Neural Networks for Age and Gender Classification". In: *Proceedings of the ICCV'17 Workshop on Analysis and Modeling of Faces and Gestures (AMFG)*.

Lapuschkin, S., S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller (2018). "Assessing What Machines Really Learn – Unmasking 'Clever Hans' Predictors". In: *Nature Communications*. In Revision.

Lazebnik, S., C. Schmid, and J. Ponce (2006). "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. IEEE, pp. 2169–2178.

LeCun, Y. (1998). *The MNIST Database of Handwritten Digits*. http://yann.lecun.com/exdb/mnist/.

LeCun, Y., L. Bottou, G. B. Orr, and K.-R. Müller (2012). "Efficient BackProp". In: *Neural Networks: Tricks of the Trade, Reloaded. vol. 7700 of Lecture Notes in Computer Science (LNCS)*. Springer, pp. 9–48.

LeCun, Y., K. Kavukcuoglu, and C. Farabet (2010). "Convolutional Networks and Applications in Vision". In: *International Symposium on Circuits and Systems (ISCAS 2010), May 30 - June 2, 2010, Paris, France*, pp. 253–256.

Levi, G. and T. Hassner (2015). "Age and Gender Classification using Convolutional Neural Networks". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops, Boston, MA, USA, June 7-12, 2015*, pp. 34–42.

Lin, M., Q. Chen, and S. Yan (2014). "Network In Network". In: *International Conference of Learning Representations (ICLR)*.

Linderman, G. C. and S. Steinerberger (2017). "Clustering with t-SNE, Provably". In: *CoRR* abs/1706.02582. arXiv: 1706.02582.

Linsley, D., S. Eberhardt, T. Sharma, P. Gupta, and T. Serre (2017). "What are the Visual Features Underlying Human Versus Machine Vision?" In: *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017*, pp. 2706–2714.

Lipton, Z. C. (2018). "The Mythos of Model Interpretability". In: *ACM Queue* 16.3, p. 30.

Liu, L., L. Wang, and X. Liu (2011). "In Defense of Soft-assignment Coding". In: *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pp. 2486–2493.

Lou, Y., R. Caruana, and J. Gehrke (2012). "Intelligible Models for Classification and Regression". In: *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pp. 150–158.

Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2, pp. 91–110.

Maaten, L. v. d. and G. Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.Nov, pp. 2579–2605.

MacQueen, J. et al. (1967). "Some Methods for Classification and Analysis of Multivariate Observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA., pp. 281–297.

Mahendran, A. and A. Vedaldi (2015). "Understanding Deep Image Representations by Inverting Them". In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Maji, S., A. C. Berg, and J. Malik (2008). "Classification using Intersection Kernel Support Vector Machines is Efficient". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, pp. 1–8.

MATLAB (2016). *version 9.1.0 (R2010b)*. Natick, Massachusetts: The MathWorks Inc.

McCulloch, W. S. and W. Pitts (1943). "A Logical Calculus of the Ideas Immanent in Nervous Activity". In: *The bulletin of mathematical biophysics* 5.4, pp. 115–133.

Meila, M. and J. Shi (2001). "A Random Walks View of Spectral Segmentation". In: *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics, AISTATS 2001, Key West, Florida, US, January 4-7, 2001*.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, et al. (2015). "Human-level Control through Deep Reinforcement Learning". In: *Nature* 518.7540, pp. 529–533.

Montavon, G., S. Bach, A. Binder, W. Samek, and K.-R. Müller (2016). "Deep Taylor Decomposition of Neural Networks". In: *Proceedings of the ICML'16 Workshop on Visualization for Deep Learning*, pp. 1–3.

Montavon, G., M. L. Braun, T. Krueger, and K.-R. Müller (2013a). "Analyzing Local Structure in Kernel-based Learning: Explanation, Complexity, and Reliability Assessment". In: *IEEE Signal Processing Magazine* 30.4, pp. 62–74.

Montavon, G., M. L. Braun, and K.-R. Müller (2010). "Layer-wise Analysis of Deep Networks with Gaussian Kernels". In: *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada.* Pp. 1678–1686.

— (2011). "Kernel Analysis of Deep Networks". In: *Journal of Machine Learning Research* 12.Sep, pp. 2563–2581.

Montavon, G., S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller (2017). "Explaining NonLinear Classification Decisions with Deep Taylor Decomposition". In: *Pattern Recognition* 65, pp. 211–222.

Montavon, G., M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld (2013b). "Machine Learning of Molecular Electronic Properties in Chemical Compound Space". In: *New Journal of Physics* 15.9, p. 095003.

Montavon, G., W. Samek, and K.-R. Müller (2018). "Methods for Interpreting and Understanding Deep Neural Networks". In: *Digital Signal Processing* 73, pp. 1–15.

Moosmann, F., E. Nowak, and F. Jurie (2008). "Randomized Clustering Forests for Image Classification". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.9, pp. 1632–1646.

Moravčík, M., M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. H. Bowling (2017). "DeepStack: Expert-Level Artificial Intelligence in No-Limit Poker". In: *CoRR* abs/1701.01724. arXiv: 1701.01724.

Müller, K.-R., S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf (2001). "An Introduction to Kernel-based Learning Algorithms". In: *IEEE Trans. Neural Networks* 12.2, pp. 181–201.

Müller, K.-R., A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik (1997). "Predicting Time Series with Support Vector Machines". In: *Artificial Neural Networks - ICANN '97, 7th International Conference, Lausanne, Switzerland, October 8-10, 1997, Proceedings*, pp. 999–1004.

Ng, A. Y., M. I. Jordan, and Y. Weiss (2002). "On Spectral Clustering: Analysis and an Algorithm". In: *Advances in Neural Information Processing Systems*, pp. 849–856.

Nguyen, A. M., J. Yosinski, and J. Clune (2015). "Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pp. 427–436.

Nowak, S., K. Nagel, and J. Liebetrau (2011). "The CLEF 2011 Photo Annotation and Concept-based Retrieval Tasks". In: *CLEF 2011 Labs and Workshop, Notebook Papers, 19-22 September 2011, Amsterdam, The Netherlands*.

Oquab, M., L. Bottou, I. Laptev, and J. Sivic (2014). "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pp. 1717–1724.

O'toole, A. J., T. Vetter, N. F. Troje, and H. H. Bülthoff (1997). "Sex Classification is Better with Three-dimensional Head Structure than with Image Intensity Information". In: *Perception* 26.1, pp. 75–84.

Parliament and Council of the European Union (2016). "General Data Protection Regulation". In:

Parra, L. C., C. D. Spence, A. D. Gerson, and P. Sajda (2005). "Recipes for the Linear Analysis of EEG". In: *Neuroimage* 28.2, pp. 326–341.

Pearson, K. (1901). "LIII. On Lines and Planes of Closest Fit to Systems of Points in Space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11, pp. 559–572.

Perotin, L., R. Serizel, E. Vincent, and A. Guérin (2018). "CRNN-based Multiple DoA Estimation Using Ambisonics Acoustic Intensity Features". In:

Perronnin, F., J. Sánchez, and T. Mensink (2010). "Improving the Fisher Kernel for Large-Scale Image Classification". In: *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*, pp. 143–156.

Phinyomark, A., G. Petri, E. Ibáñez-M arcelo, S. T. Osis, and R. Ferber (2017). "Analysis of Big Data in Gait Biomechanics: Current Trends and Future Directions". In: *Journal of Medical and Biological Engineering*, pp. 1–17.

Pinto, N. and J. J. Cox David D.and DiCarlo (2008). "Why is Real-World Visual Object Recognition Hard?" In: *PLoS Computational Biology* 4.1.

Pörner, N., B. Roth, and H. Schütze (2018). "Evaluating Neural Network Explanation Methods Using Hybrid Documents and Morphosyntactic Agreement". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 340–350.

Quellec, G., K. Charrière, Y. Boudi, B. Cochener, and M. Lamard (2017). "Deep Image Mining for Diabetic Retinopathy Screening". In: *Medical Image Analysis* 39, pp. 178–193.

Rasmus, A., M. Berglund, M. Honkala, H. Valpola, and T. Raiko (2015). "Semi-supervised Learning with Ladder Networks". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3546–3554.

Rasmussen, P. M., T. Schmah, K. H. Madsen, T. E. Lund, S. C. Strother, and L. K. Hansen (2012). *Visualization of Nonlinear Classification Models in Neuroimaging*.

Reyes, E., P. A. Estévez, I. Reyes, G. Cabrera-Vives, P. Huijse, R. Carrasco-Davis, and F. Förster (2018). "Enhanced Rotational Invariant Convolutional Neural Network for Supernovae Detection". In: *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018*, pp. 1–8.

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

Rosenblatt, F. (1957). *The Perceptron, a Perceiving and Recognizing Automaton*. Cornell Aeronautical Laboratory.

Rothe, R., R. Timofte, and L. Van Gool (2016). "Deep Expectation of Real and Apparent Age from a Single Image Without Facial Landmarks". In: *International Journal of Computer Vision*, pp. 1–14.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). "Learning Representations by Back-propagating Errors". In: *Nature* 323, pp. 533–536.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. (2015). "Imagenet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3, pp. 211–252.

Samek, W., A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller (2017). "Evaluating the visualization of what a Deep Neural Network has learned". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.11, pp. 2660–2673.

Samek, W., G. Montavon, A. Binder, S. Lapuschkin, and K.-R. Müller (2016). "Interpreting the Predictions of Complex ML Models by Layer-wise Relevance Propagation". In: *Proceedings of the Interpretable ML for Complex Systems NIPS'16 Workshop*, pp. 1–5.

Sánchez, J., F. Perronnin, T. Mensink, and J. J. Verbeek (2013). "Image Classification with the Fisher Vector: Theory and Practice". In: *International Journal of Computer Vision* 105.3, pp. 222–245.

Sande, K. E. A. van de, T. Gevers, and C. G. M. Snoek (2010). "Evaluating Color Descriptors for Object and Scene Recognition". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 32.9, pp. 1582–1596.

Schirrmeister, R. T., J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball (2017). "Deep Learning with Convolutional Neural Networks for Brain Mapping and Decoding of Movement-related Information from the Human EEG". In: *CoRR* abs/1703.05051. arXiv: 1703.05051.

Schölkopf, B., C. J. Burges, and A. Smola (1998a). "Advances in Kernel Methods Support Vector Learning". In: *Cambridge, MA, MIT Press*.

Schölkopf, B., R. Herbrich, and A. J. Smola (2001). "A Generalized Representer Theorem". In: *Computational Learning Theory, 14th Annual Conference on Computational Learning Theory, COLT 2001 and 5th European Conference on Computational Learning Theory, EuroCOLT 2001, Amsterdam, The Netherlands, July 16-19, 2001, Proceedings*, pp. 416–426.

Schölkopf, B., A. J. Smola, and K.-R. Müller (1998b). "Nonlinear Component Analysis as a Kernel Eigenvalue Problem". In: *Neural Computation* 10.5, pp. 1299–1319.

Schölkopf, B., R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt (1999). "Support Vector Method for Novelty Detection". In: *Advances in Neural Information Processing Systems 12, NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pp. 582–588.

Schütt, K. T., F. Arbabzadah, S. Chmiela, K.-R. Müller, and A. Tkatchenko (2017). "Quantum-chemical Insights from Deep Tensor Neural Networks". In: *Nature Communications* 8, p. 13890.

Schütt, K. T., M. Kloft, A. Bikadorov, and K. Rieck (2012). "Early detection of malicious behavior in JavaScript code". In: *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence, AISec 2012, Raleigh, NC, USA, October 19, 2012*, pp. 15–24.

Schütt, K. T., H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller (2018). "SchNet–A Deep Learning Architecture for Molecules and Materials". In: *The Journal of Chemical Physics* 148.24, p. 241722.

Schwenk, G. and S. Bach (2014). "Detecting Behavioral and Structural Anomalies in MediaCloud Applications". In: *CoRR* abs/1409.8035. arXiv: 1409.8035.

Seibold, C., W. Samek, A. Hilsmann, and P. Eisert (2018). "Accurate and Robust Neural Networks for Security Related Applications Exampled by Face Morphing Attacks". In: *CoRR* abs/1806.04265. arXiv: 1806.04265.

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra (2017). "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 618–626.

Shi, L., Z. Teng, L. Wang, Y. Zhang, and A. Binder (2018). "DeepClue: Visual Interpretation of Text-based Deep Stock Prediction". In: *IEEE Transactions on Knowledge and Data Engineering*.

Shimodaira, H. (2000). "Improving Predictive Inference under Covariate Shift by Weighting the Log-likelihood Function". In: *Journal of statistical planning and inference* 90.2, pp. 227–244.

Shrikumar, A., P. Greenside, and A. Kundaje (2017). "Learning Important Features Through Propagating Activation Differences". In: *CoRR* abs/1704.02685. arXiv: 1704.02685.

Shrikumar, A., P. Greenside, A. Shcherbina, and A. Kundaje (2016). "Not Just a Black Box: Learning Important Features Through Propagating Activation Differences". In: *CoRR* abs/1605.01713. arXiv: 1605.01713.

Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, et al. (2016). "Mastering the Game of Go with Deep Neural Networks and Tree Search". In: *Nature* 529.7587, pp. 484–489.

Simoncelli, E. P. and B. A. Olshausen (2001). "Natural Image Statistics and Neural Representation". In: *Annual review of neuroscience* 24.1, pp. 1193–1216.

Simonyan, K., A. Vedaldi, and A. Zisserman (2013). "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps". In: *CoRR* abs/1312.6034. arXiv: 1312.6034.

Simonyan, K. and A. Zisserman (2014). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556. arXiv: 1409.1556.

Slijepcevic, D., M. Zeppelzauer, A.-M. Gorgas, C. Schwab, M. Schüller, A. Baca, C. Breiteneder, and B. Horsak (2018). "Automatic Classification of Functional Gait Disorders". In: *IEEE J. Biomedical and Health Informatics* 22.5, pp. 1653–1661.

Sommer, R. and V. Paxson (2010). "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection". In: *31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berleley/Oakland, California, USA*, pp. 305–316.

Sonnenburg, S., G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. De Bona, A. Binder, C. Gehl, and V. Franc (2010). "The SHOGUN Machine Learning Toolbox". In: *Journal of Machine Learning Research* 11, pp. 1799–1802.

Spille, C. and B. T. Meyer (2017). "Listening in the Dips: Comparing Relevant Features for Speech Recognition in Humans and Machines". In: *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pp. 2968–2972.

Springenberg, J. T., A. Dosovitskiy, T. Brox, and M. A. Riedmiller (2015). "Striving for Simplicity: The All Convolutional Net". In: *International Conference of Learning Representations (ICLR)*.

Srinivasan, V., S. Lapuschkin, C. Hellge, K.-R. Müller, and W. Samek (2017). "Interpretable Human Action Recognition in Compressed Domain". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1692–1696.

Steinwart, I., D. R. Hush, and C. Scovel (2006). "An Explicit Description of the Reproducing Kernel Hilbert Spaces of Gaussian RBF Kernels". In: *IEEE Trans. Information Theory* 52.10, pp. 4635–4643.

Sturm, I., S. Lapuschkin, W. Samek, and K.-R. Müller (2016). "Interpretable Deep Neural Networks for Single-Trial EEG Classification". In: *Journal of Neuroscience Methods* 274, pp. 141–145.

Swain, M. J. and D. H. Ballard (1991). "Color Indexing". In: *International journal of computer vision* 7.1, pp. 11–32.

Szegedy, C., S. Ioffe, V. Vanhoucke, and A. A. Alemi (2017). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning." In: 4, p. 12.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015). "Going Deeper with Convolutions". In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9.

Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna (2016). "Rethinking the Inception Architecture for Computer Vision". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826.

Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus (2013). "Intriguing Properties of Neural Networks". In: *CoRR* abs/1312.6199. arXiv: 1312.6199.

Tax, D. M. J. and R. P. W. Duin (2004). "Support Vector Data Description". In: *Machine Learning* 54.1, pp. 45–66.

Thomas, A. W., H. R. Heekeren, K.-R. Müller, and W. Samek (2018). "Interpretable LSTMs For Whole-Brain Neuroimaging Analyses". In: *CoRR* abs/1810.09945. arXiv: 1810.09945.

Tsotsos, J. K., S. M. Culhane, W. Y. K. Wai, Y. Lai, N. Davis, and F. Nuflo (1995). "Modeling Visual Attention via Selective Tuning". In: *Artif. Intell.* 78.1-2, pp. 507–545.

Uijlings, J. R. R., A. W. M. Smeulders, and R. J. H. Scha (2012). "The Visual Extent of an Object - Suppose We Know the Object Locations". In: *International Journal of Computer Vision* 96.1, pp. 46–63.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer.

Vedaldi, A. and B. Fulkerson (2010). "VlFeat: An Open and Portable Library of Computer Vision Algorithms". In: *Proceedings of the 18th International Conference on Multimedia 2010, Firenze, Italy, October 25-29, 2010*, pp. 1469–1472.

Vedaldi, A., V. Gulshan, M. Varma, and A. Zisserman (2009). "Multiple Kernels for Object Detection". In: *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pp. 606–613.

Vidovic, M. M.-C., N. Görnitz, K.-R. Müller, G. Rätsch, and M. Kloft (2015). "SVM2Motif — Reconstructing Overlapping DNA Sequence Motifs by Mimicking an SVM Predictor". In: *PloS one* 10.12, e0144782.

Von Luxburg, U. (2007). "A Tutorial on Spectral Clustering". In: *Statistics and Computing* 17.4, pp. 395–416.

Wahba, G. (1990). *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics.

Wang, J., J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong (2010). "Locality-constrained Linear Coding for Image Classification". In: *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pp. 3360–3367.

Wang, S. I. and C. D. Manning (2012). "Baselines and Bigrams: Simple, Good Sentiment and Topic Classification". In: *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*, pp. 90–94.

Xiao Jianxiongand Hays, J., A. Ehinger Krista A.and Oliva, and A. Torralba (2010). "SUN Database: Large-scale Scene Recognition from Abbey to Zoo". In: *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pp. 3485–3492.

Yang, J., K. Yu, Y. Gong, and T. S. Huang (2009). "Linear Spatial Pyramid Matching using Sparse Coding for Image Classification". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 1794–1801.

Yang, Y., V. Tresp, M. Wunderle, and P. A. Fasching (2018). "Explaining Therapy Predictions with Layer-Wise Relevance Propagation in Neural Networks". In: *IEEE International Conference on Healthcare Informatics, ICHI 2018, New York City, NY, USA, June 4-7, 2018*, pp. 152–162.

Yosinski, J., J. Clune, A. M. Nguyen, and H. Fuchs Thomas J.and Lipson (2015). "Understanding Neural Networks Through Deep Visualization". In: *CoRR* abs/1506.06579. arXiv: 1506.06579.

Yu, K., T. Zhang, and Y. Gong (2009). "Nonlinear Learning using Local Coordinate Coding". In: *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.* Pp. 2223–2231.

Zagoruyko, S. and S. Komodakis (2016). "Wide Residual Networks". In: *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*.

Zeiler, M. D. and R. Fergus (2014). "Visualizing and Understanding Convolutional Networks". In: *Proc. of European Conference on Computer Vision (ECCV)*. Springer, pp. 818–833.

Zhang Ningand Donahue, J., R. B. Girshick, and T. Darrell (2014). "Part-Based R-CNNs for Fine-Grained Category Detection". In: *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, pp. 834–849.

Zhang, J., M. Marszalek, S. Lazebnik, and C. Schmid (2007). "Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study". In: *International Journal of Computer Vision* 73.2, pp. 213–238.

Zhang, J., Z. Bargal Sarah Adeland Lin, J. Brandt, X. Shen, and S. Sclaroff (2018). "Top-Down Neural Attention by Excitation Backprop". In: *International Journal of Computer Vision* 126.10, pp. 1084–1102.

Zhang, X., J. J. Zhao, and Y. LeCun (2015). "Character-level Convolutional Networks for Text Classification". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 649–657.

Zhou, B., A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba (2014a). "Object Detectors Emerge in Deep Scene CNNs". In: *CoRR* abs/1412.6856. arXiv: 1412.6856.

— (2016). "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*, pp. 2921–2929.

Zhou, B., À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva (2014b). "Learning Deep Features for Scene Recognition using Places Database". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 487–495.

Zhu, Y., Y. Zhou, Q. Ye, and J. Qiu Qiangand Jiao (2017). "Soft Proposal Networks for Weakly Supervised Object Localization". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 1859–1868.

Zien, A., N. Krämer, S. Sonnenburg, and G. Rätsch (2009). "The Feature Importance Ranking Measure". In: *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part II*, pp. 694–709.

Zintgraf, L. M., T. S. Cohen, T. Adel, and M. Welling (2017). "Visualizing Deep Neural Network Decisions: Prediction Difference Analysis". In: *International Conference on Learning Representations (ICLR), 2017*.

Zintgraf, L. M., T. S. Cohen, and M. Welling (2016). "A New Method to Visualize Deep Neural Networks". In: *CoRR* abs/1603.02518. arXiv: 1603.02518.