Separable explanations of neural network decisions

Laura Rieger
DTU Compute
Denmark Technical University
lauri@dtu.dk

Abstract

Deep Taylor Decomposition is a method used to explain neural network decisions. When applying this method to non-dominant classifications, the resulting explanation does not reflect important features for the chosen classification. We propose that this is caused by the dense layers and propose a method to alleviate the effect by applying regularization. We assess the result by measuring the quality of the resulting explanations objectively and subjectively.

1 Introduction

In recent years, Neural Networks (NNs) have achieved great success in many applications, particularly in the visual domain [1, 2, 3]. However, they essentially function as a black box for the decision they made. This is undesirable for a number of applications. Especially for applications like medical diagnosis and other heavily influential decisions it is essential that we can verify and argue for a decision and knowing the reasoning is as important as the decision itself [4, 5, 6].

Deep Taylor Decomposition is a recently proposed approach to retrieve an explanation for a classification [7]. When applying this method to classes other than the class with maximum output, the explanation does not reflect the features related to the non-dominant classification. We call this effect diffusion, as the explanation for non-dominant classes is diffused by the explanation for the dominant class. For this reason, Deep Taylor Decomposition usually fails the purpose of explaining multiple classifications. However, this information is important for some applications. We want to know what evidence speaks for a less dominant class when we do risk analysis, when we want to merge information from different sources or to analyze an NN for finding weaknesses.

2 Related Work

2.1 Other approaches to explainability

Recently a number of papers have dealt with explaining NNs [8, 9, 10, 7]. They can be categorized into following one of two basic approaches. Either the goal is to find general properties of the NN or obtain an explanation for a specific decision. In this work we concentrate on the latter.

Simonyan et al. [9] computes a saliency map for an input image in regards to a specific class score. By finding pixels of the image where the class score has a high gradient, they identify important areas of the image. The method works for all classes recognized by the network equally well without considering saliency for other classes. However, as argued in multiple works, this method only visualizes what would change the class score but not what caused it [11, 12, 13, 7]. Bach et al. [12] proposed a method for explaining an output score in terms of pixels of the input. In particular, it introduces a Taylor decomposition that finds a near root point, where the output score would be zero, and performs expansion approximation from this point along with iterative relevance propagation to retrieve the relevance distribution on the input.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

2.2 Deep Taylor Decomposition

In this section we describe Deep Taylor Decomposition (DTD), the method for explaining NN decisions introduced in Montavon et al. [7]. A more extensive explanation can be found in Montavon et al. [14].

DTD assigns relevance for the classification to each discrete part of the input [7]. Subsequently, we will only talk about images where pixels are the discrete input. We want to find a mapping $R_p(\boldsymbol{x})$, that assigns relevance for the NN classification to each pixel p in the input image $\boldsymbol{x} = \{x_p\}$ with x_p denoting a specific pixel. Positive values $R_p(\boldsymbol{x}) > 0$ indicate that the pixel x_p is relevant for the classification. The result is a heatmap highlighting important parts of the image with dimensions equal to the original image as shown in Figure 1.

DTD propagates relevance layer-wise backward, similar to backpropagation during training. In the output layer $\boldsymbol{x_k} = \{x_k\}$ the relevance R_k for each neuron is the output of that neuron x_k.

For the preceding layers, the relevance R_j of each neuron x_j in layer $\boldsymbol{x_j}$ can be written as a Taylor decomposition of the neurons in the preceding layer $\boldsymbol{x_i}$ evaluated at a near root point $\tilde{\boldsymbol{x_i}}^{(j)}$ with $R_j = 0$. The relevance of each neuron x_i can then be written as the sum of their contributions to $\mathbf{R_j}$:

$$R_i = \sum_j \frac{\partial R_j}{\partial x_i}|_{\{\tilde{x}_i^{(j)}\}} \cdot (x_i - \tilde{x}_i^{(j)}) \tag{1}$$

We consider a simple layer with non-linearity $\{x_j\}$:

$$x_j = \max(0, w_{ij}x_i + b_j) \tag{2}$$

where $\{x_i\}$ is the input and $\{w_{ij}, b_j\}$ are weight and bias parameters of the layer. An additional constraint $b_j \leq 0$ is placed on the bias parameter b_j of each neuron x_j to guarantee the existence of a root point at the origin.

Since each root point should lie in the admissible input space, injecting a legal, near root point results in different propagation rules, depending on the input space. For layers following the input layer, the input space is constrained to $\mathbb{R}^+$, since ReLU is used as the activation function in the previous layer. Inserting the nearest point in this space into Equation (1) leads to the z^+-rule as defined in [7] as:

$$R_i = \sum_j \frac{x_i w_{ij}^+}{\sum_i x_i w_{ij}^+} R_j \tag{3}$$

with $w_{ij}^+ = \max(w_{ij}, 0)$. w_{ij} is the weight connecting neuron x_i to neuron x_j in the following layer.

The $z^{\mathcal{B}}$-rule as defined in [7] is used to propagate relevance from the first layer back to the input image, where the admissible input space lies between a minimum value $l_i \leq 0$ and a maximum value $h_i \geq 0$:

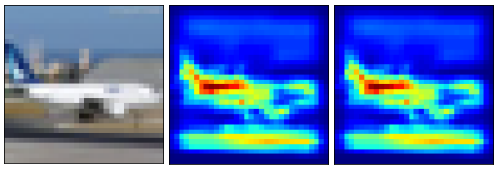
$$R_i = \sum_j \frac{x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_{i'} x_i w_{i'j} - l_i w_{i'j}^+ - h_i w_{i'j}^-} R_j \tag{4}$$

with $w_{ij}^+ = \max(0, w_{ij})$ and equivalently $w_{ij}^- = \min(0, w_{ij})$. The heatmap is summed up over all color channels for each pixel, since we are interested in the contribution of each pixel, not each color channel.

3 Diffusion of explanations

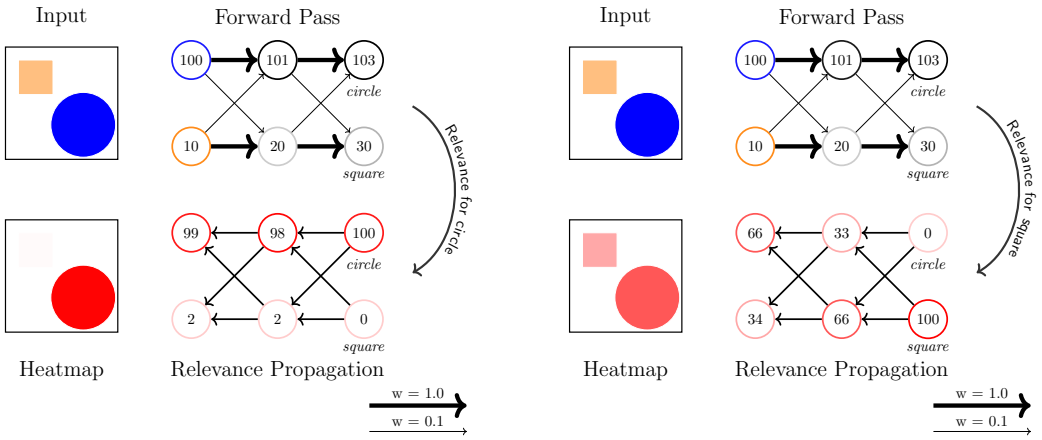
We can apply DTD to a less-dominant class by setting the output for all other classes to 0 for the relevance propagation as shown in Figure 2b. Intuitively, we expect to have features relevant for this class highlighted in the resulting heatmap, as we propagate only the output relevance for this class back. This does not happen. Instead, if DTD is applied to another class than the correct one, the resulting heatmap is very similar to the maximum output heatmap and does not reflect features speaking for the weaker class. In Figure 1 we show this for an example from CIFAR10.

We call this effect diffusion, since relevant features of the class, that we are interested in, are being diluted or diffused by highly salient features in the image. Subsequently, we will refer to an image



(a) Input (b) HM *car* (c) HM *plane*

Figure 1: Diffusion of explanation heatmaps (HM) for a sample image from CIFAR10. Heatmaps look identical for all classes.



(a) Relevance propagation for dominant class (blue circle)

(b) Relevance propagation for non-dominant class (orange square)

Figure 2: Diffusion of features when applying DTD to different classes. Explanation for non-dominant class (*square*) gets diffused with dominant explanation.

or network that has heatmaps with a higher difference for different classifications as one with low diffusion.

A potential cause for this is, that each neuron x_i is activated to reflect the dominant features in the input image since only the output was artificially changed. The relevance propagation in DTD is dependent on neuron activation in each layer. Highly activated neurons overshadow the neurons that activate for features not as dominant in the image. As a result, the relevance heatmap does not change as expected when an artificial output distribution is used.

We demonstrate this for a toy example in Figure 2. Weights are either 0.1 (thinner connection) or 1.0 (thicker connection). The input layer sums up the color intensities for the respective colors in the image, since in our toy example circles are almost always blue and squares are almost always orange. The output is a two-dimensional vector classifying the image as either *square* or *circle*. We apply the z^+ rule for backpropagation, as every input is bigger than zero.

Even though in the lower image the square should have more relevance, the highly activated neurons dominate the heatmap and the majority of relevance is attributed to the circle. We suspect that the same effect on a larger scale causes heatmaps for artificial outputs to look identical to the heatmap for the true output score when applying DTD.

4 Mitigating diffusion by block-wise layer regularization

If diffusion is caused by different neurons overshadowing each other, we can mitigate it by separating the flow of information so that neuron activations will not influence each other. Our objective is to reduce diffusion by separating the flow of information for different classes. At the same time, we want to minimize the additional training expense by minimizing the separation of layers.

NNs for image recognition are normally made up of a number of convolutional blocks, followed by multiple dense layers. Features recognized in a convolutional layer are likely to be universal for different classes, especially in the lower layers. In contrast, the combination of features in the dense layers that result in a class score is likely unique to a specific class. We therefore restrict the separation to the dense layers of an NN.

An intuitive approach would be to divide the dense layers and train a dedicated version of them for each class. This will completely separate the flow of information but highly increase the training cost, since we have to train one version of the dense layers for each class.

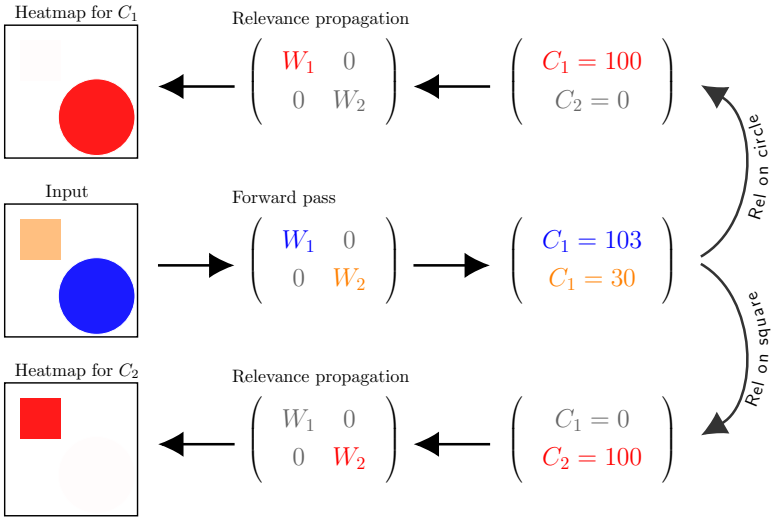


Figure 3: DTD for a layer with L1 regularization. Weights outside of the diagonal blocks are regularized to encourage sparsity.

We therefore propose softly separating the flow of information for different classifications by sparsifying the dense layers with L1 regularization. L1 regularization encourages sparsity and functions as a feature mechanism selector [15, p.236 f.].

In each dense layer, we leave a number of blocks equal to the number of classes along the diagonal unregularized. This is visualized in Figure 3. Weights, that lie outside of those blocks, are sparsified with an added L1 penalty in the loss function. Since only connections with positive weights contribute to the relevance as explained in Section 2.2, we only regularize positive weights.

As a result, only the weights connecting a specific class block in one layer to the same class block in the next layer are unregularized. We separate the flow of information for each class without increasing the number of layers or the number of neurons in each layer. To enforce L1 regularization we add the L1 penalty term to the cost function. Different from conventional L1 regularization, which penalizes all weights, only positive weights outside of the block matrices are regularized. This leads to the following equations:

$$\underset{W,b}{\text{minimize}} \quad -\frac{1}{N}\sum_{n=1}^{N} \hat{y_n}^T \cdot \log(y_n) + \lambda \cdot S(W) \tag{5}$$

$$S(W) = \sum_{l}\sum_{i}\sum_{i} \max(0, w_{ij}^{l}) \cdot (1_{i:C!=j:C}) \tag{6}$$

with w_{ij}^{l} being the weight connecting the i-th neuron in layer $l-1$ layer to the j-th neuron in layer l. λ is the regularizer rate. C is the number of classes. y_n is the NN output for a particular input and $\hat{y}_n$ is the corresponding ground truth. The weight is regularized if it is positive and connects two neurons from different class blocks, i.e. when applying integer division with the number of classes to the neuron indices results in different values.

5 Experiments

We apply our method to NNs trained with different datasets to assess the viability and report results for regularizing with varying values of λ.

5.1 Setup

We show results for using the L1 regularization on CIFAR10 [16], MNIST [17] and FashionMNIST [18].

Table 1: Used network architectures

Dataset	L1	L2	L3	L4	L5	L6	L7	Output
CIFAR10	Cov(32,3,3)	Cov(32,3,3)	MaxP	Cov(64,3,3)	Cov(64,3,3)	MaxP	D(512)	D(10)
FASHION	Cov(32,3,3)		AvgP				D(256)	D(10)
MNIST	Cov(32,3,3)		AvgP		Cov(64,2,2)	AvgP	D(512)	D(10)

For clarity, we show results for one network architecture for each dataset. In Table 1 the used network architectures are shown. Dropout was applied during training to prevent overfitting. For NNs trained for CIFAR10 we additionally used horizontal flipping and shifting for data augmentation during training. All networks were pre-trained without regularization until the loss no longer improved for three concurrent epochs. This pre-trained network is used as a baseline. This NN is fine-tuned by training with L1 regularization as described in Section 4 until loss has no longer improved for three epochs. We fine-tuned networks with $\lambda \in \{0.1, 0.5, 1.0, 5.0\}$.

To assess the viability of the proposed method we need a way to measure diffusion and its reduction. Ideally, we would have an objective metric for the meaningfulness of an explanation in regards to a particular class. Unfortunately, it is not possible to obtain a ground truth for this. Therefore, we use a combination of two measures. To objectively measure the diffusion for one specific NN, we take the Frobenius norm of the difference between heatmaps for the two classes with the highest output for one image. We average this value over a number of images from the test set, resulting in:

$$d_k = \frac{1}{N} \sum_{n \in N} \|R_{n,c_1} - R_{n,c_2}\|_F \tag{7}$$

with $c_{\{1,2\}}$ being the class with the highest or second-highest output respectively. $R_{n,c}$ represents the relevance distribution at the input level for image n and class c. We average the diffusion over Nimages. Diffusion between two versions of the same network architecture can be compared by comparing the average diffusion values. In this way we can assess the effectiveness of regularization in regards to reducing diffusion.

This objectively measures the difference between explanations for different classes, which we want to increase. We visually assess the heatmaps for test images to additionally judge whether the difference is meaningful.

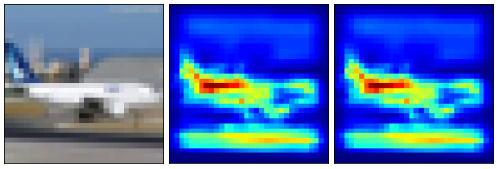
5.2 Results

Table 2: Diffusion metric when applying soft separation with varying strength of separation with λ

λ	CIFAR10	FASHION	MNIST
Baseline	0.13	0.42	0.28
0.1	0.12	0.76	**1.06**
0.5	**0.59**	1.80	0.98
1.0	0.58	**1.85**	0.97
5.0	0.56	1.72	1.02

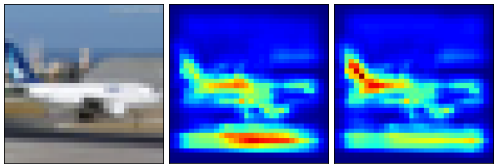
In Table 2 we show the difference metric defined in Section 5.1 averaged over randomly sampled images from the test set for the baseline network and networks regularized with varying λ. We see, that applying L1 regularization almost always results in decreased diffusion and more differences between heatmaps. The λ-value at which the lowest diffusion is reached varies. Since the value

correlates with the final accuracy and final loss of the network, a likely explanation is that the same regularization value does not influence the training as much if the other component of the loss-function is also high.



Input HM *Car* HM *Plane*

(a) Baseline NN



Input HM *Car* HM *Plane*

(b) L1 regularized NN

Figure 4: Baseline NN (upper row) and L1 regularized NN (lower row) heatmap comparison for CIFAR10 image. The L1 regularized network puts more relevance on the wing for *plane*. In the HM for *car*, the street background is more relevant.

In Figures 4 and 5 we show heatmaps for test images created by a baseline NN and the $\lambda = 1.0$ regularized version. It is visible that the difference in the explanation from the L1 regularized NN aligns with our intuitive explanation, i.e. the wings of a plane are more relevant for class *plane* than class *car*. Diffusion is still existent, though reduced. As shown in Table 3, the accuracy does not

Table 3: Accuracy when applying soft separation with varying strength of separation with λ compared to the baseline

λ	CIFAR10	FASHION	MNIST
Baseline	0.80	0.91	0.99
0.1	0.80	0.91	0.99
0.5	0.81	0.91	0.99
1.0	0.81	0.91	0.99
5.0	0.81	0.91	0.99

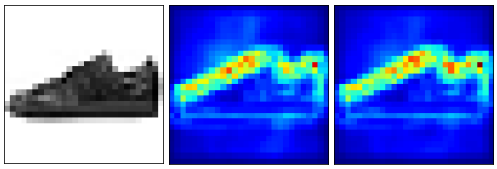
decrease notably when regularizing the network. Computation time per epoch does not increase significantly, although fine-tuning the NN adds more total computation time, since we need to train the NN for additional epochs. This time could be further reduced by freezing the lower convolutional layers during fine-tuning.

6 Discussion

Separating the dense layers led to less diffused heatmaps for different classes. This indicates that diffusion is at least partially caused by high neuron activations overshadowing each other. We suspect, that another cause is only considering positive weights during the relevance propagation, as DTD does not take features speaking against a class into account.

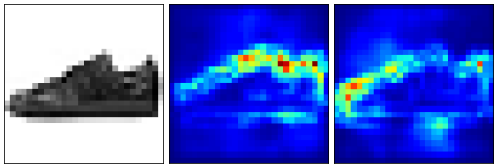
We introduced a way to measure diffusion, dependent on the difference to the heatmap for the original output. Ideally, we would measure how much the heatmap created is different from the ground truth. Since we do not have a ground truth for the heatmap, measuring the difference offers a viable alternative if we ensure that heatmaps are produced in a meaningful way.

Constraining the dense layers with an L1 penalty led to a measurable reduction of diffusion in heatmaps while not harming accuracy of the NN. We found that the explanation for single classes



Input HM *Sneaker* HM *Ankle boot*

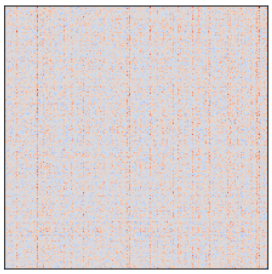
(a) Baseline NN



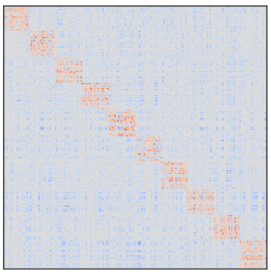
Input HM *Sneaker* HM *Ankle boot*

(b) L1 regularized NN

Figure 5: Baseline NN and L1 regularized NN heatmap comparison on FashionMNIST. For the L1 regularized network, the *sneaker* ending below the ankle is more relevant compared to the *ankle boot*.



(a) Baseline



(b) L1 regularized ($\lambda = 1.0$)

Figure 6: Visualization of dense layer weights for baseline and L1 regularized network. Positive values are red, negative values are blue.

improves by using a pre-trained neural net and fine-tuning with L1-regularization without suffering a loss in accuracy. In Figure 6, a visualization of a densely connected layer, it is also visible that the positive weights are largely restricted to the respective class blocks and will therefore not influence each other during relevance propagation. A disadvantage is that we add an additional free parameter.

In addition to retrieving explanations for non-dominant classes, it would follow that soft block regularization leads to more meaningful explanations for the dominant class as well, since heatmaps no longer include features that add relevance to other classes in highly ambiguous images. However, we cannot verify this in a quantified way.

7 Conclusion

The goal of our work was to retrieve sensible explanations for non-dominant classifications with DTD. We find that enforcing sparsity in the dense layers alleviates diffusion and leads to more meaningful explanations. Using L1 regularization to softly separate dense layers in fine-tuning leads to a sensible explanation for non-dominant classes. It reduces diffusion but does not eliminate it. Unfortunately, this solution is not scalable for a large number of classes, as each class would not have enough unrestricted neurons for a good classification performance. Solving this problem for DTD remains an open problem.

Acknowledgments

References

[1] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

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

[4] Julia AngwinJeff LarsonLauren KirchnerSurya Mattu. Machine bias, May 2016. URL `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`.

[5] Bryce Goodman and Seth Flaxman. Eu regulations on algorithmic decision-making and a "right to explanation". In *ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, 2016.

[6] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

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

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

[9] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. URL `http://dblp.uni-trier.de/db/journals/corr/corr1312.html#SimonyanVZ13`.

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

[11] Will Landecker, Michael D Thomure, Luís MA Bettencourt, Melanie Mitchell, Garrett T Kenyon, and Steven P Brumby. Interpreting individual classifications of hierarchical networks. In *Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on*, pages 32–38. IEEE, 2013.

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

[13] Jianming Zhang, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. In *European Conference on Computer Vision*, pages 543–559. Springer, 2016.

[14] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 2017.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT Press, 2016.

[16] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[17] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

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