A Rate-Distortion Framework for Explaining Deep Neural Network Decisions

Stephan Waeldchen*, Jan Macdonald*, Sascha Hauch*, and Gitta Kutyniok*
*Technische Universität Berlin, Institut für Mathematik, Berlin, Germany
Emails: stephanw@math.tu-berlin.de, macdonald@math.tu-berlin.de, hauch@math.tu-berlin.de, kutyniok@math.tu-berlin.de

***Abstract*—We propose a rate-distortion framework for explaining deep neural network decisions. We formulate the task of determining the most relevant input signal components for a classifier prediction as an optimisation problem. For the special case of binary signals and Boolean classifier functions we show it to be $\mathsf{NP}^{\mathsf{PP}}$-complete as well as NP-hard to approximate. Finally, we present a heuristic solution strategy based on assumed density filtering and tailored specifically to the case of deep ReLU neural network classifiers for which we present numerical experiments.**

I. Introduction

Recently machine learning techniques in general and deep neural networks in particular have been successfully applied to a variety of problems such as image classification [1], objection recognition [2], or speech recognition [3]. Although these techniques empirically achieve impressive predictive accuracy the theoretical foundation for their success is often lacking. An increasing interest in utilising deep learning methods also for high-stakes applications, such as medical imaging and diagnosis, in which reliable predictions are of crucial importance, have led to a growing desire to understand and interpret the predictions and decisions made by these methods and obtain information about their uncertainty.

Over the last years several methods for estimating uncertainties in neural networks [4] as well as methods to interpret their decisions [5], [6] have been proposed. Here, we introduce a rigorous approach to obtaining interpretable neural networks. More precisely, in section II we formulate the problem of determining the most relevant components of an input signal for a classifier prediction as an optimisation problem in a rate-distortion framework. We show in section III that this problem is generally hard to solve and to approximate, which justifies the use of heuristic methods. In section IV we propose a problem relaxation together with a heuristic solution strategy tailored to our rate-distortion formulation of the problem, and finally present numerical experiments in section V.

A. Notation

Throughout the paper $d \in \mathbb{N}$ denotes the dimension of the signal domain, $\mathbf{x} \in [0,1]^d$ is an arbitrary fixed input signal and $\Phi\colon [0,1]^d \to [0,1]$ is a classifier function for a signal class $\mathcal{C} \subseteq [0,1]^d$. The function Φ can for example be described by a neural network. The classification score $\Phi(\mathbf{x})$ represents the classifiers prediction on how likely it is that $\mathbf{x}$ belongs to the class $\mathcal{C}$. We denote $[d] = \{1, \dots, d\}$ and for a subset $S \subseteq [d]$ denote by $\mathbf{x}_S$ the restriction of $\mathbf{x}$ to components indexed by S. Finally, $\mathbb{1} \in \mathbb{R}^d$ denotes a vector of all ones, $\operatorname{diag}(\mathbf{x})$ the diagonal matrix with entries given by $\mathbf{x}$, and $\odot$ (resp. $\oslash$) the component-wise Hadamard product (resp. quotient) of two vectors or matrices of the same dimensions. For simplicity we write $\mathbf{x}^2 = \mathbf{x} \odot \mathbf{x}$. Throughout, univariate functions applied to vectors will be considered to act component-wise.

II. Rate-Distortion Viewpoint

The task is to find a subset $S \subseteq [d]$ of *relevant* components of $\mathbf{x}$ and its complement S^c of *non-relevant* components such that fixing the relevant components already determines the output of the classifier for almost all possible assignments to the non-relevant components. More precisely, let $\mathcal{V}$ be a probability distribution on the domain $[0,1]^d$ and $\mathbf{n} \sim \mathcal{V}$ a random vector. We define the *obfuscation* of $\mathbf{x}$ with respect to S and $\mathcal{V}$ as a random vector $\mathbf{y}$ that is deterministically defined on S as $\mathbf{y}_S = \mathbf{x}_S$ and distributed on the complement according to $\mathbf{y}_{S^c} = \mathbf{n}_{S^c}$. We write $\mathcal{V}_S$ for the distribution of $\mathbf{y}$, keeping the dependence on $\mathbf{x}$ implicit. The expected distortion of S with respect to Φ, $\mathbf{x}$, and $\mathcal{V}$ is defined as

$$D(S) = D(S, \Phi, \mathbf{x}, \mathcal{V}) = \mathbb{E}_{\mathbf{y} \sim \mathcal{V}_S}\left[\frac{1}{2}\left(\Phi(\mathbf{x}) - \Phi(\mathbf{y})\right)^2\right]. \quad (1)$$

As Φ, $\mathbf{x}$, and $\mathcal{V}$ are fixed throughout this paper we will simply use $D(S)$ from now on.

We naturally arrive at a rate-distortion trade-off that intuitively gives us a measure of relevance. We define the rate-distortion function as

$$R(\epsilon) = \min\{\, |S| \,:\, S \subseteq [d],\, D(S) \leq \epsilon \,\}, \quad (2)$$

again implicitly dependent on $\mathbf{x}$, $\mathcal{V}$ and Φ. The smallest set S that ensures a limited distortion will be composed of the most relevant input components.

We now want to discuss the difficulty of finding such a set. Note, that the trivial choice of setting $S = [d]$ ensures zero distortion. We show that for distortion limits greater than zero one cannot systematically find a set of relevant components that is significantly smaller than the trivial set. This even holds when we restrict the problem to the set of functions defined as neural networks which is the class of functions that we are particularly interested in.

A. Neural Network Functions

Let $L \in \mathbb{N}$, $d_1, \dots, d_{L-1} \in \mathbb{N}$ and denote $d_0 = d$, $d_L = 1$. Further let $(\mathbf{W}_1, \mathbf{b}_1), \dots, (\mathbf{W}_L, \mathbf{b}_L)$ with $\mathbf{W}_i \in \mathbb{R}^{d_i \times d_{i-1}}$,

$\mathbf{b}_i \in \mathbb{R}^{d_i}$ for $i \in [L]$ be the weight matrices and bias vectors of a L-layer neural network. We then consider functions of the form

$$\Phi(\mathbf{x}) = \mathbf{W}_L \varrho(\mathbf{W}_{L-1}\varrho(\dots \varrho(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)\dots) + \mathbf{b}_{L-1}) + \mathbf{b}_L$$

where the activation function $\varrho\colon \mathbb{R} \to \mathbb{R}$ is applied component-wise. A commonly used activation function is the rectified linear unit (ReLU) activation $\varrho(x) = \max\{0, x\}$.

We will see that our hardness results hold for the special case of Boolean circuits that can be represented by ReLU neural networks of moderate size[1].

III. Complexity Theoretic Analysis

Let us for now consider the special case of binary input signals $\mathbf{x} \in \{0,1\}^d$ and functions $\Phi : \{0,1\}^d \to \{0,1\}$ described as Boolean circuits, as well as the uniform distribution over binary vectors, i.e. $\mathcal{V} = \mathcal{U}\left(\{0,1\}^d\right)$.

In that case for any realisation of the obfuscation $\mathbf{y}$ the difference $\Phi(\mathbf{x}) - \Phi(\mathbf{y})$ can either be zero or one. Hence, we can give a simple counting formulation of (1) that allows us to classify the hardness of the problem. Define the sets

$$A_S := \left\{\mathbf{y} \in \{0,1\}^d \,:\, \mathbf{x}_S = \mathbf{y}_S\right\},$$
$$B_S := \left\{\mathbf{y} \in \{0,1\}^d \,:\, \mathbf{x}_S = \mathbf{y}_S \text{ and } \Phi(\mathbf{x}) = \Phi(\mathbf{y})\right\}.$$

The set B_S is simply the subset of A_S satisfying the constraint $\Phi(\mathbf{x}) = \Phi(\mathbf{y})$.

Definition 1. We call S a *δ-relevant* set for Φ and $\mathbf{x}$, with $\delta \in (0,1]$, if $\frac{|B_S|}{|A_S|} \geq \delta$.

The task of calculating the size of the smallest δ-relevant set is the same as calculating the rate $R(\epsilon)$ for a maximal distortion of $\epsilon = \frac{1}{2}(1-\delta)$.

Definition 2. The Relevant-Input problem is:
Given: $\Phi : \{0,1\}^d \to \{0,1\}$, $\mathbf{x} \in \{0,1\}^d$, $\delta \in (0,1]$, $k \in [d]$.
Decide: Does there exist an $S \subseteq [d]$ with $|S| \leq k$ such that S is δ-relevant for Φ and $\mathbf{x}$?

The optimisation version of this problem, called Minimal Relevant-Input, asks for the smallest k, such that there exists a δ-relevant set of size at most k. The following two hardness results hold.

Theorem 3. Relevant-Input *is* $\mathsf{NP}^{\mathsf{PP}}$*-complete.*

The class $\mathsf{NP}^{\mathsf{PP}}$ is the class of all problems decidable by a non-deterministic Touring machine equipped with an oracle for problems in PP [7]. The class $\mathsf{NP}^{\mathsf{PP}}$ appears frequently in artificial intelligence tasks, such as optimisation under uncertainty and is assumed significantly harder than either NP and PP. A more practically relevant result is the following.

Theorem 4. *Assume* $\mathsf{P} \neq \mathsf{NP}$*, then for any* $\alpha \in (0,1)$ *there is no polynomial time approximation algorithm for* Minimal Relevant-Input *with an approximation factor of* $d^{1-\alpha}$.

[1]meaning that the depth L can be chosen constant and the total number of neurons $\sum_i d_i$ bounded by a polynomial in d.

For the proofs see [8] (in preparation). Theorem 4 shows that no efficient approximation algorithm exists (unless $\mathsf{P} = \mathsf{NP}$). Either we have to resort to heuristics, or introduce stronger restrictions on the problem. We choose the former and present a general heuristic for neural networks.

IV. Problem Relaxation and Solution Heuristic

In section III we considered Boolean circuit functions $\Phi\colon \{0,1\}^d \to \{0,1\}$ and saw that Minimal Relevant-Input is hard to solve and to approximate. Thus, we further relax the problem.

A. Continuous Problem Relaxation

Instead of binary relevance decisions (*relevant* versus *non-relevant*) encoded by the set S, we allow for a continuous relevance score for each component, encoded by the vector $\mathbf{s} \in [0,1]^d$. We redefine the obfuscation of $\mathbf{x}$ with respect to $\mathbf{s}$ as a component-wise convex combination

$$\mathbf{y} = \mathbf{x} \odot \mathbf{s} + \mathbf{n} \odot (\mathbb{1} - \mathbf{s}) \tag{3}$$

of $\mathbf{x}$ and $\mathbf{n} \sim \mathcal{V}$. As before we write $\mathcal{V}_\mathbf{s}$ for the distribution of $\mathbf{y}$. This is a generalisation of the obfuscation introduced in section II which can be recovered by choosing $\mathbf{s}$ equal to one on S and zero on S^c. The natural relaxation of the size of the set S is given by the norm $\|\mathbf{s}\|_1$. Analogous to before we define the distortion

$$D(\mathbf{s}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{V}_\mathbf{s}}\left[\frac{1}{2}\left(\Phi(\mathbf{x}) - \Phi(\mathbf{y})\right)^2\right].$$

Instead of the hard constraint $D(\mathbf{s}) \leq \epsilon$ as in (2) we formulate the continuous rate minimisation problem in its Lagrangian formulation

$$\begin{aligned} &\text{minimize} && D(\mathbf{s}) + \lambda\|\mathbf{s}\|_1 \\ &\text{subject to} && \mathbf{s} \in [0,1]^d \end{aligned} \tag{4}$$

with a regularisation parameter $\lambda > 0$. Depending on the activation function, the distortion does not need to be differentiable. However, the ReLU activation is differentiable almost everywhere. As commonly done during the training of neural networks, we simply use (projected) gradient descent to find a stationary point of (4).

The exact calculation of expectation values for arbitrary functions is in itself already a hard problem. One possibility to overcome this issue is to approximate the expectation by a sample mean and use batched stochastic gradient descent, similar to the way neural networks are trained. Here however, we focus on a second possibility, which takes the specific structure of Φ more into account.

B. Assumed Density Filtering

We utilise the layered structure of Φ and propagate the distribution of the neuron activations through the network. No family of distributions is invariant under the transformation of a neural network layer, except for a mixture of δ-distributions which amounts to the sampling approach discussed above.

Instead we use an approximate method, called assumed density filtering (ADF), see for example [9], [10], which has recently also been used for ReLU neural networks in the context of uncertainty quantification [11]. In a nutshell, at each layer we assume a Gaussian distribution for the input, transform it according to the layers weights $\mathbf{W}$, biases $\mathbf{b}$, and activation function ϱ, and project the output back to the nearest Gaussian distribution (w.r.t. KL-divergence). This amounts to matching the first two moments of the distribution [9].

Using the bias-variance decomposition of the mean squared error, we can rewrite

$$D(\mathbf{s}) = \frac{1}{2}\left(\Phi(\mathbf{x}) - \mathbb{E}_{\mathbf{y}\sim\mathcal{V}_\mathbf{s}}\left[\Phi(\mathbf{y})\right]\right)^2 + \frac{1}{2}\mathbb{V}_{\mathbf{y}\sim\mathcal{V}_\mathbf{s}}\left[\Phi(\mathbf{y})\right].$$

The distortion is determined by the first and second moment of the output layer. From (3) it is straight-forward to obtain the first and second moment of $\mathbf{y}$ depending on $\mathbf{s}$. We now derive the ADF rules to propagate them through the network layers to get an explicit expression for the distortion.

Let $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. An affine linear transformation acts on the mean and covariances in the well-known way

$$\mathbb{E}\left[\mathbf{W}\mathbf{z} + \mathbf{b}\right] = \mathbf{W}\boldsymbol{\mu} + \mathbf{b}, \tag{5}$$

$$\mathbb{V}\left[\mathbf{W}\mathbf{z} + \mathbf{b}\right] = \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^*, \tag{6}$$

where $\mathbb{V}$ denotes the covariance matrix.

The ReLU non-linearity ϱ presents a difficulty as it changes a Gaussian distribution into a non-Gaussian one. Let f and F be the probability density and cumulative distribution function of the univariate standard normal distribution, let $\boldsymbol{\sigma}$ be the vector of the diagonal entries of $\boldsymbol{\Sigma}$, and $\boldsymbol{\eta} = \boldsymbol{\mu} \oslash \boldsymbol{\sigma}$. Then

$$\mathbb{E}\left[\varrho(\mathbf{z})\right] = \boldsymbol{\sigma} \odot f(\boldsymbol{\eta}) + \boldsymbol{\mu} \odot F(\boldsymbol{\eta}).$$

Unfortunately there is no closed form expression for the off-diagonal terms of the covariance matrix of $\varrho(\mathbf{z})$. We either make the additional assumption that the network activations in each layer are uncorrelated, which amounts to propagating only the diagonal $\mathbb{V}_{\text{diag}}$ of the covariance matrices through the network and simplifies (6) to

$$\mathbb{V}_{\text{diag}}\left[\mathbf{W}\mathbf{z} + \mathbf{b}\right] = \left(\mathbf{W} \odot \mathbf{W}\right)\boldsymbol{\sigma},$$

and results in

$$\mathbb{V}_{\text{diag}}\left[\varrho(\mathbf{z})\right] = \boldsymbol{\mu}\odot\boldsymbol{\sigma}\odot f(\boldsymbol{\eta}) + \left(\boldsymbol{\sigma}^2 + \boldsymbol{\mu}^2\right)\odot F(\boldsymbol{\eta}) - \mathbb{E}\left[\varrho(\mathbf{z})\right]^2.$$

Or we use an approximation for the full covariance matrix

$$\mathbb{V}\left[\varrho(\mathbf{z})\right] \approx \mathbf{N}\boldsymbol{\Sigma}\mathbf{N},$$

with $\mathbf{N} = \operatorname{diag}\left(F(\boldsymbol{\eta})\right)$. This approximation ensures positive semi-definiteness. Altogether, combining the linear transformation with the non-linear one, tells us how to propagate the first two moments through a ReLU neural network layer in the ADF framework. We investigate both the *diagonal* as well as the *full* covariance matrix method in our numerical inquiry.

V. Numerical Experiments

We present a numerical experiment for interpretable neural networks comparing our proposed method to several other widely used techniques. We generate relevance mappings for greyscale images of handwritten digits from the MNIST dataset [12]. An example image from the dataset can be seen in fig. 1 (top-left).

The focus of this paper is on the interpretability of neural networks, not on their training so we will keep the description of the training process quite brief.

The methods we compare to are: Layer-wise Relevance Propagation (LRP) [5], Deep Taylor Decompositions [6], Sensitivity Analysis, the Input $\times$ Gradient method, and simply taking the input signal itself as a relevance map. Different interpretation approaches produce differently scaled and normalised relevance mappings which makes it hard to directly compare them. Some methods generate non-negative mappings corresponding to the importance *for* the classifier score, whereas other methods also generate negative relevances that can be interpreted as speaking *against* a classifier decision.

To allow for a fair comparison of the methods we propose a variant of the *relevance ordering*-based test introduced in [13]. Each relevance mapping induces an ordering of the input signal components by sorting them according to their relevance score (breaking ties randomly). Starting with the least relevant pixels, we then replace increasingly large parts of the input signal by random components and observe the change in the classifier score. A good relevance mapping will lead to slow changes in the classifier score when the most relevant components are replaced last.

We use Tensorflow/Keras for our experiments. All comparison relevance mappings are generated using Innvestigate [14].

For the experiment we trained a simple six-layer convolutional neural network to classify greyscale images of handwritten digits using the MNIST dataset [12]. We trained for 80 epochs with mini-batches of size 128.

An example for the relevance mappings generated by our as well as the comparison methods is shown in fig. 1. We restricted the neural network output to the class with the highest prediction score before the softmax normalisation. We observe that both variants of our method generate almost identical relevance mappings. The top-right area of the image discriminating the digit six from, for example, an eight or a zero is mostly highlighted by our method. The relevance ordering based comparison test is shown in fig. 3. We see that our proposed method results in a classifier score that remains almost constant until about the 90% least relevant input components have been randomised. Samples from the comparison test with the 60% least relevant components randomised are visualised in fig. 2.

Acknowledgements

The authors would like to thank Philipp Petersen for several fruitful discussions during the early stage of the project. S. W. and J. M. acknowledge support by DFG-GRK-2260 (BIOQIC). S. H. is grateful for support by CRC/TR 109 “Discretization in

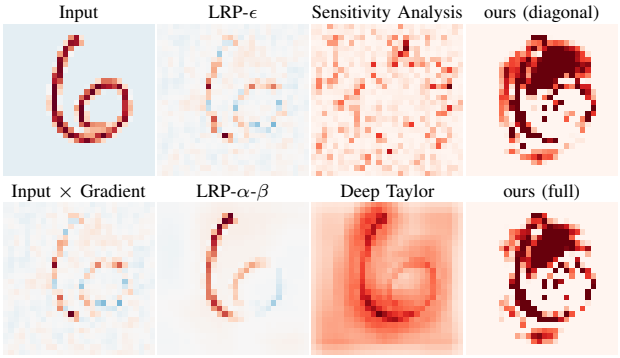

Figure 1. Relevance mappings generated by several methods for an image from the MNIST dataset classified as *digit six* by our network. The colourmap indicates positive relevances as red and negative relevances as blue.

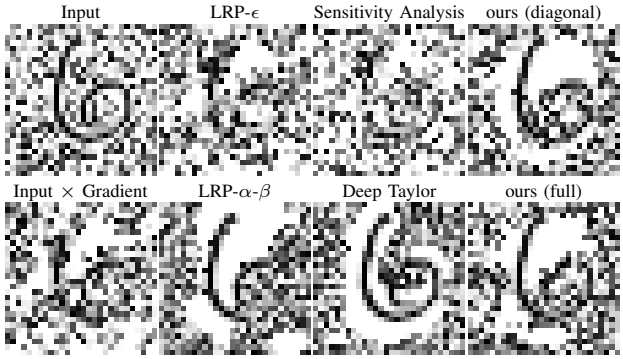

Figure 2. Samples from the relevance ordering comparison test for an image from the MNIST dataset with the 60% least relevant input components randomised.

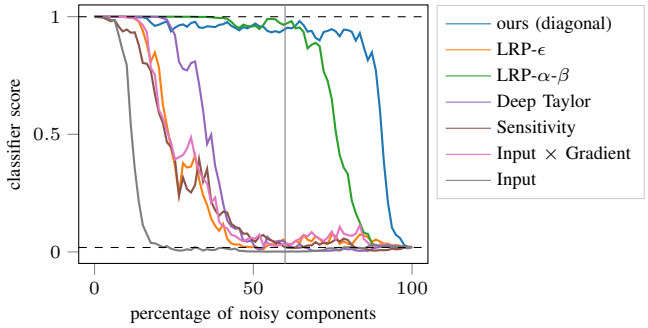

Figure 3. Relevance ordering based comparison test for relevance mappings generated by several methods for an image from the MNIST dataset. The most relevant components are randomised last. Classifier scores for the original input signal and the completely randomised image are shown as horizontal dashed lines for reference. The vertical line marks 60% randomisation (cf. fig. 2).

Geometry and Dynamics". G. K. acknowledges partial support by the Bundesministerium für Bildung und Forschung (BMBF) through the Berliner Zentrum for Machine Learning (BZML), Project AP4, by the Deutsche Forschungsgemeinschaft (DFG) through Grants CRC 1114 "Scaling Cascades in Complex Systems", Project B07, CRC/TR 109 "Discretization in Geometry and Dynamics", Projects C02 and C03, RTG DAEDALUS (RTG 2433), Projects P1 and P3, RTG BIOQIC (RTG 2260), Projects P4 and P9, SPP 1798 "Compressed Sensing in Information Processing", Project Coordination and Project Massive MIMO-I/II, by the Berlin Mathematics Research Center MATH+, Projects EF1-1 and EF1-4, and by the Einstein Foundation Berlin.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2553–2561. [Online]. Available: http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf

[3] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 05 2013, pp. 6645–6649.

[4] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 6402–6413. [Online]. Available: http://papers.nips.cc/paper/7219-simple-and-scalable-predictive-uncertainty-estimation-using-deep-ensembles.pdf

[5] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLOS ONE*, vol. 10, no. 7, pp. 1–46, 07 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0130140

[6] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[7] J. Gill, "Computational complexity of probabilistic turing machines," *SIAM Journal on Computing*, vol. 6, no. 4, pp. 675–695, 1977.

[8] S. Waeldchen, J. Macdonald, S. Hauch, and G. Kutyniok, "The computational complexity of understanding network decisions," in preparation.

[9] T. P. Minka, "A family of algorithms for approximate bayesian inference," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 2001, aAI0803033.

[10] X. Boyen and D. Koller, "Tractable inference for complex stochastic processes," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 33–42. [Online]. Available: http://dl.acm.org/citation.cfm?id=2074094.2074099

[11] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 3369–3378.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[13] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 11, pp. 2660–2673, 11 2017.

[14] M. Alber, S. Lapuschki, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K. Müller, S. Dähne, and P. Kindermans, "innvestigate neural networks!" *CoRR*, vol. abs/1808.04260, 2018. [Online]. Available: http://arxiv.org/abs/1808.04260