A Rate-Distortion Framework for Explaining Deep Neural Network Decisions

Jan Macdonald*, Stephan Waeldchen*, Sascha Hauch*, and Gitta Kutyniok*

*Technische Universität Berlin, Institut für Mathematik, Berlin, Germany
Emails: macdonald@math.tu-berlin.de, stephanw@math.tu-berlin.de, hauch@math.tu-berlin.de, kutyniok@math.tu-berlin.de

I. INTRODUCTION

Recently deep neural networks have been successfully applied to a variety of problems such as image classification [1], object recognition [2], or speech recognition [3]. This led to a growing desire to understand and interpret the predictions and decisions made by these networks. Several ways to interpret their decisions have been proposed [4, 5]. Here, we introduce a rigorous approach to obtaining interpretable neural networks. 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 and show that this problem is generally hard to solve and to approximate, which justifies the use of heuristic methods. Further, we propose a problem relaxation together with a heuristic solution strategy tailored to our rate-distortion formulation of the problem, and finally present some numerical experiments.

II. RATE-DISTORTION VIEWPOINT

Let \( \Phi : [0, 1]^d \rightarrow [0, 1] \) be a classifier function (e.g. described by a neural network) for a signal class \( \mathcal{C} \subseteq [0, 1]^d \) and \( x \in [0, 1]^d \) a fixed input signal. The classification score \( \Phi(x) \) represents the classifier prediction on how likely it is that \( x \) belongs to \( \mathcal{C} \). The task is to find a subset \( S \subseteq \{1, \ldots, d\} \) of relevant components of \( x \) such that fixing the relevant components already determines the output of the classifier for almost all possible assignments to the non-relevant components in \( S^c \). For this define the obfuscation of \( x \) with respect to \( S \) and a probability distribution \( V \) on \( [0, 1]^d \) as a random vector \( y \) with \( y_S = x_S \) and \( y_{S^c} = n_{S^c} \) on \( S^c \) with \( n \sim V \). We write \( V_S \) for the distribution of \( y_S \). Throughout, \( E \) and \( V \) will denote expected value and variance of random variables. The expected distortion of \( S \) with respect to \( \Phi, x, V \) and \( V \) is defined as

\[
D(S) = D(S, \Phi, x, V) = E_{y \sim V_S} \left[ \frac{1}{2} (\Phi(x) - \Phi(y))^2 \right].
\]

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

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

implicitly dependent on \( x, V \) and \( \Phi \). The smallest set \( S \) that ensures a limited distortion will be composed of the most relevant input components.

III. COMPLEXITY THEORETIC ANALYSIS

Let us now consider the special case of binary inputs \( x \in \{0, 1\}^d \), Boolean circuit functions \( \Phi : \{0, 1\}^d \rightarrow \{0, 1\} \), and the uniform distribution \( V = U\{\{0, 1\}^d\} \). In this case the computational complexity of determining the rate-distortion function as well as approximating it can be analysed. See [6] (in preparation) for details.

**Theorem 1.** Assume \( \Phi \neq NP \), then for any \( \alpha \in \{0, 1\} \) there is no polynomial time approximation algorithm that computes \( R(\epsilon) \) with an approximation factor of \( d^{1-\alpha} \) for all Boolean circuits \( \Phi \), binary \( x \), and \( \epsilon > 0 \).

IV. PROBLEM RELAXATION AND SOLUTION HEURISTIC

Let us get back to the general setting with \( \Phi : [0, 1]^d \rightarrow [0, 1], x \in [0, 1]^d \), and arbitrary \( V \). 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 \( s \in [0, 1]^d \). We redefine the obfuscation of \( x \) with respect to \( s \) as a component-wise convex combination

\[
y = x \odot s + n \odot (1 - s)
\]

of \( x \) and \( n \sim V \). As before we write \( V_s \) for the distribution of \( y \). We replace the size \( S \) by \( \|s\|_1 \), and arrive at the Lagrangian formulation of the continuous rate-distortion minimisation problem

\[
\text{minimize } D(s) + \lambda \|s\|_1
\]

subject to \( s \in [0, 1]^d \),

where the distortion can be written in its bias-variance decomposition

\[
D(s) = \frac{1}{2} (\Phi(x) - \mathbb{E}_{y \sim V_s} [\Phi(y)])^2 + \frac{1}{2} \mathbb{V}_{y \sim V_s} [\Phi(y)],
\]

and \( \lambda > 0 \) is a regularisation parameter. The distortion is determined by the first and second moment of \( \Phi(y) \). From [1] it is straightforward to obtain the first and second moment of \( y \) depending on \( s \) and \( V \). In the case where \( \Phi \) is a multi-layer neural network we can estimate the moments of \( \Phi(y) \) by approximately propagating the probability distribution of \( y \) through the network layers. For this we use the assumed density filtering approach as described in [7], which in a nutshell propagates Gaussian distributions through the network layers, projecting back to the best (moment matching) Gaussian after each layer. We investigate both propagating the full covariance matrix and only its diagonal elements which is numerically faster. We then solve the approximated rate minimisation problem simply by (projected) gradient descent.

V. NUMERICAL EXPERIMENTS

For numerical evaluation, we trained a ReLU neural network classifier (three convolution layers each followed by average-pooling and finally two fully-connected layers) on the MNIST dataset [8] up to a test accuracy of 0.99. We calculated the relevance maps for individual digits according to our method as well as several widely used techniques (fig. [1]) taken from [2]. For a fair comparison we propose a variant of the relevance ordering-based test introduced in [10]. We order the pixel from least to most relevant and randomise each technique (fig. [2]). A good relevance map leads to a slow decay of the classifier score over the percentage of randomised pixels. As shown in fig. [3], our proposed method outperforms the other methods in this test scenario.
Figure 1. Relevance mappings generated by several methods for an image from the MNIST dataset classified as *digit six* by our neural network. The colour map indicates positive relevances as red and negative relevances as blue.

Figure 2. Samples from the relevance ordering comparison test for an image (cf. fig 1) 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 (cf. fig 1) from the MNIST dataset. For the test we replace an increasingly large percentage of components of the input signal (ordered by their relevance score) by samples from i.i.d uniform noise in the interval $[0,1]$ and observe changes in the classifier score (averaged over 100 random samples for each percentage of noisy components). If the pixels were ordered correctly, the important pixels are randomised last and the decay in classification accuracy is slow. 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).