Interpreting Deep Neural Networks Through Variable Importance

Jonathan Ish-Horowicz[*]
Imperial College London

Dana Udwin[*]
Brown University

Seth Flaxman
Imperial College London

Lorin Crawford[†]
Brown University

Sarah Filippi[†]
Imperial College London

Abstract

While the success of deep neural networks is well-established across a variety of domains, our ability to explain and interpret these methods is limited. Unlike previously proposed *local* methods which try to explain particular classification decisions, we focus on *global* interpretability and ask a universally applicable, and surprisingly understudied question: given a trained model, which features are the most important? In the context of neural networks, a feature is rarely important on its own, so our strategy is specifically designed to leverage partial covariance structures and incorporate variable interactions into our proposed feature ranking. Our methodological contributions in this paper are three-fold. First, we propose a novel effect size analogue for the problem of global interpretability, which is appropriate for applications with highly collinear predictors (ubiquitous in computer vision). Second, we extend the recently proposed "RelATive cEntrality" (RATE) measure [1] to the Bayesian deep learning setting. RATE applies an information theoretic criterion to the posterior distribution of effect sizes to assess feature significance. Unlike competing methods, our method has no tuning parameters to pick or costly randomization steps. Finally, we propose a novel extension, groupRATE, to assess the importance of specified groups of variables. Overall, we show state-of-the-art results applying our framework to several application areas including: computer vision, genetics, natural language processing, and social science.

1 Introduction

Due to their high predictive performances, deep neural networks (DNNs) have become increasingly used in many fields including computer vision and natural language processing. Unfortunately, DNNs operate as "black boxes": users are rarely able to understand the internal workings of the network. As a result, DNNs have not been widely adopted in scientific settings, where variable selection tasks are often as important as prediction — one particular example being the identification of biomarkers related to the progression of a disease. While DNNs are beginning to be used in high-risk decision-making fields (e.g. automated medical diagnostics or self-driving cars), it is critically important that methods do not make predictions based on artifacts or biases in the training data. Therefore, there is both a strong theoretical and practical motivation to increase the global interpretability of DNNs, and to better characterize the types of relationships upon which they rely.

Despite being an increasingly important concept in machine learning, interpretability lacks a well-established definition in the literature. Such inconsistencies have lead to a lack of consensus on how

[*]Equal contribution

[†]Corresponding Authors: lorin_crawford@brown.edu; s.filippi@imperial.ac.uk

Preprint. Under review.

interpretability should be achieved or evaluated. Variable importance is one possible approach to achieve global interpretability, where the goal is to rank each input feature based on its contributions to predictive accuracy. This is in contrast to local interpretability, which aims to simply provide an explanation behind a specific prediction or group of predictions. In this paper, we follow a more recently proposed definition which refers to interpretability as "the ability to explain or to present in understandable terms to a human" [2]. To this end, our main contribution is focused on global interpretability; we address the problem of identifying important predictor variables given a trained neural network, focusing especially on settings in which variables or groups of variables are intrinsically meaningful.

Here, we describe an approach to achieve global interpretability for deep neural networks using "RelATive cEntrality" (RATE) [1], a recently-proposed variable importance criterion for Bayesian models. This flexible approach can be used with any network architecture where some notion of uncertainty can be computed over the predictions. The rest of the paper is structured as follows. Section 2 outlines related work on the interpretation of deep neural networks. Section 3 describes the RATE computation within the context for which it was originally proposed (Gaussian process regression). Section 4 contains the main methodological innovations of this paper. Here, we present a unified framework under which RATE can be applied to deep neural networks and propose groupRATE for ranking groups of variables. In Section 5, we demonstrate the utility of our method in various simulation scenarios and real data applications, and compare to competing approaches.

2 Related Work

In the absence of a robustly defined metric for interpretability, most work on DNNs has focused on methods that can be evaluated visually, and especially on local interpretability—trying to explain specific classification decisions with respect to input features. In this work, we focus instead on global interpretability where the goal is to identify predictor variables that best explain the overall performance of a trained model. Previous work in this context has focused on selecting inputs that maximize the activation of each layer within the network [3]. Another viable approach for achieving global interpretability is to train more conventional statistical methods to mimic the predictive behavior of a DNN. This imitation or *mimic* model is then retrospectively used to explain the predictions that a DNN would make. For example, using a decision tree [4] or falling rule list [5] can yield straightforward characterizations of predictive outcomes. Unfortunately, these simple models can struggle to mimic the accuracy of DNNs effectively. A random forest (RF) or gradient boosting machine (GBM), on the other hand, is much more capable of matching the predictive power of DNNs. Measures of feature importance can be computed for RFs and GBMs by permuting information within the input variables and examining this null effect on test accuracy, or by calculating Gini impurity [6]. The ability to establish variable importance in random forests is a significant reason for their popularity in fields such as the life and clinical sciences [7], where random forest and gradient boosting machine mimic models have been used as interpretable predictive models for patient outcomes [8]. A notable drawback of RFs and GBMs is that it can take a significant amount of training time to achieve accuracy comparable to the DNNs that they serve to mimic. This provides motivation for our direct approach, avoiding the need to train a separate model.

3 Relevant Background

In this section, we give a brief review on previous results that are relevant to our main methodological innovations. Throughout, we assume access to some trained model, with the ability to draw samples from its posterior predictive distribution. This reflects the *post-hoc* nature of our objective of finding important subsets of variables. Assume that we have an n-dimensional response vector $\mathbf{y}$ and an $n \times p$ design matrix $\mathbf{X}$ with p covariates. For linear models, an effect size is defined as the projection of the response onto the column space of the data $\mathbf{X}^\dagger \mathbf{y}$, with $\mathbf{X}^\dagger$ being the Moore-Penrose pseudo-inverse. In the Bayesian nonparametric setting, we consider a learned nonlinear function that has been evaluated on the n-observed samples, where $\mathbb{E}[\mathbf{y} \mid \mathbf{X}] = \boldsymbol{f}$. The *effect size analogue* can then be defined as the result of projecting the vector $\boldsymbol{f}$ onto the original design matrix $\mathbf{X}$,

$$\widetilde{\boldsymbol{\beta}} = \text{Proj}(\mathbf{X}, \boldsymbol{f}). \tag{1}$$

Some intuition can be gained as follows. After having fit a model, we consider the fitted values $\boldsymbol{f}$ and regress these predictions onto the input variables so as to see how much variance these features explain.

This is a simple way of understanding the relationships that the model has learned. The coefficients produced by this linear projection have their normal interpretation: they provide a summary of the relationship between the covariates in $\mathbf{X}$ and $\boldsymbol{f}$. For example, while holding everything else constant, increasing some feature $\mathbf{x}_j$ by 1 will increase $\boldsymbol{f}$ by $\widetilde{\beta}_j$. In the case of kernel machines, theoretical results for identifiability and sparsity conditions of the effect size analogue have been previously developed when using the Moore-Penrose pseudo-inverse as a projection operator [9].

Similar to regression coefficients in linear models, effect size analogues are not used to solely determine variable significance. Indeed, there are many approaches to infer associations based on the magnitude of effect size estimates, but many of these techniques rely on arbitrary thresholding and fail to account for key covarying relationships that exist within the data. The "RelATive cEntrality" measure (or RATE) was developed as a *post-hoc* approach for variable selection that mitigates these concerns [1].

Consider a sample from the predictive distribution of $\widetilde{\boldsymbol{\beta}}$, obtained by transforming draws from the posterior of $\boldsymbol{f}$ via the deterministic projection in Eq. (1)) The RATE criterion summarizes how much any one variable contributes to what the model has learned. Effectively, this is done by taking the Kullback-Leibler divergence (KLD) between (i) the conditional posterior predictive distribution $p(\widetilde{\boldsymbol{\beta}}_{-j} \,|\, \widetilde{\beta}_j = 0)$ with the effect of the j-th predictor being set to zero, and (ii) the marginal distribution $p(\widetilde{\boldsymbol{\beta}}_{-j})$ with the effects of the j-th predictor being integrated out. Namely, $\text{RATE}(\widetilde{\beta}_j) := \text{KLD}(\widetilde{\beta}_j) / \sum_\ell \text{KLD}(\widetilde{\beta}_\ell)$ where

$$\text{KLD}(\widetilde{\beta}_j) := \text{KL}\left(p(\widetilde{\boldsymbol{\beta}}_{-j}) \,\|\, p(\widetilde{\boldsymbol{\beta}}_{-j} \,|\, \widetilde{\beta}_j = 0)\right) = \int \log\left(\frac{p(\widetilde{\boldsymbol{\beta}}_{-j})}{p(\widetilde{\boldsymbol{\beta}}_{-j} \,|\, \widetilde{\beta}_j = 0)}\right) p(\widetilde{\boldsymbol{\beta}}_{-j}) \, \mathrm{d}\widetilde{\boldsymbol{\beta}}_{-j}. \quad (2)$$

Note that $\text{RATE}(\widetilde{\beta}_j)$ is a non-negative quantity, and equals zero if and only if variable j is of little importance, since removing its effect has no influence on the other variables. In addition, the RATE criterion is bounded within the range $[0, 1]$, with the natural interpretation of measuring a variable's relative entropy — with a higher value equating to more importance. To this end, $1/p$ is a practical threshold for characterizing a predictor as "significant" since it represents the value at which the influence of all variables is uniform and indistinguishable. To build further intuition, formal links between RATE and mutual information can also be established (see Appendix 10).

4 Methodological Contributions

We now detail the main methodological contributions of this paper. First, we describe our motivating deep neural network architecture. Next, we propose a new effect size analogue projection that is more robust to collinear input data. Lastly, we derive a closed-form solution for the RATE methodology under this new framework and describe novel extensions.

4.1 Motivating Neural Network Architecture

Consider a binary classification problem with n observations. We have an n-dimensional set of labels $\mathbf{y} \in \{0, 1\}^n$ and an $n \times p$ design matrix $\mathbf{X}$ with p covariates. We assume the following hierarchical network architecture to learn the predicted label for each observation in the data

$$\widehat{\mathbf{y}} = \sigma(\boldsymbol{f}), \quad \boldsymbol{f} = \mathbf{H}(\boldsymbol{\theta})\mathbf{w} + \mathbf{b}, \quad \mathbf{w} \sim \pi(\bullet)\,, \quad (3)$$

where $\sigma(\bullet)$ is a sigmoid function, and $\boldsymbol{f}$ is an n-dimensional vector of smooth latent values or "logits" that need to be estimated. Here, we use an $n \times k$ matrix $\mathbf{H}(\boldsymbol{\theta})$ to denote the activations from the penultimate layer (which are fixed given a set of inputs and point estimates of the inner layer parameters $\boldsymbol{\theta}$), $\mathbf{w}$ is a k-dimensional vector of weights at the output layer assumed to follow prior distribution $\pi(\bullet)$, and $\mathbf{b}$ is an n-dimensional vector of the deterministic bias that is produced during the training phase.

The structure of Eq. (3) is motivated by the fact that we are most interested in the posterior distribution of the latent variables when computing the effect size analogues and, subsequently, RATE measures. To this end, we may logically split the network into three components: (i) an input layer of the original predictor variables, (ii) hidden layers where parameters are deterministically computed, and (iii) the logit layer where the parameters and activations are treated as random variables. Since the resulting

logits are a linear combination of these components, their joint distribution will be closed-form if the posterior distribution of the weight parameters is also of closed-form.

There are two important features of this network setup. First, we may easily generalize this architecture to the multi-class problem by increasing the number of output nodes to match the number of categories, and redefining $\sigma(\bullet)$ to be the softmax function. Regression is even simpler: we let $\sigma(\bullet)$ be the identity. Second, the structure of the hidden layers can be of any size or type, provided that the additional parameters are given by point estimates. Ultimately, this flexibility means that a wide range of existing architectures can be easily modified to be used with RATE. The simplest example of such an architecture is illustrated in Figure 1.

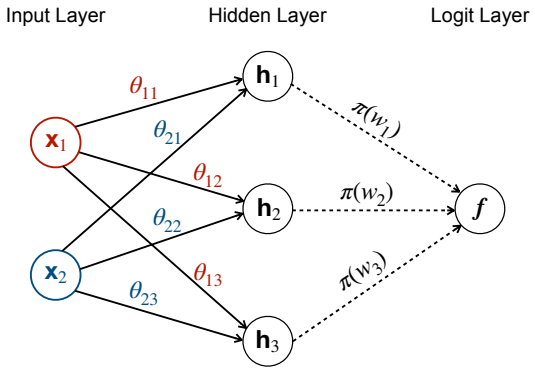

Figure 1: An example of the network architecture used in this work. The first layer parameters $\boldsymbol{\theta}$ are computed deterministically, while the logit layer weights $\mathbf{w}$ are assumed to be distributed under the prior $\pi(\bullet)$. The input variables $\mathbf{x_1}$ and $\mathbf{x_2}$ are fed through the hidden layers $(\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3)$. Estimates of the predicted logits $\boldsymbol{f}$ are obtained via a linear combination of these components and samples from the posterior distribution of (w_1, w_2, w_3). This figure does not include the deterministic bias terms.

4.2 Posterior Inference with Variational Bayes

As the size of datasets in many application areas continues to grow, it has become less feasible to implement traditional Markov Chain Monte Carlo (MCMC) algorithms for inference. This has motivated approaches for supervised learning that are based on variational Bayes and the stochastic optimization of a variational lower bound [10–12]. In this work, we use variational Bayes because it has the additional benefit of providing closed-form expressions for the posterior distribution of $\mathbf{w}$ — and, subsequently, the logits $\boldsymbol{f}$. Here, we first specify a prior $\pi(\mathbf{w})$ over the weights and replace the intractable true posterior $p(\mathbf{w} \mid \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{w})\pi(\mathbf{w})$ with an approximating family of distributions $q_{\boldsymbol{\phi}}(\mathbf{w})$. The variational parameters $\boldsymbol{\phi}$ are selected by minimizing $\mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{w}) \,\|\, p(\mathbf{w} \mid \mathbf{y})\right)$, with respect to $\boldsymbol{\phi}$, with the goal of selecting the member of the approximating family that is closest to the true posterior. This is equivalent to maximizing the so-called variational lower bound.

Since the architecture specified in Eq. (3) contains point estimates at the hidden layers, we cannot train the network by simply maximizing the lower bound with respect to the variational parameters. Instead, all parameters must be optimized jointly as follows:

$$\underset{\boldsymbol{\phi},\boldsymbol{\theta}}{\arg\max} - \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{w}) \,\|\, \pi(\mathbf{w})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{w})}\left[\log p(\mathbf{y} \mid \mathbf{w}, \boldsymbol{\theta})\right]. \tag{4}$$

We will then use stochastic optimization to train the network. Depending on the chosen variational family, the gradients of the minimized $\mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{w}) \,\|\, \pi(\mathbf{w}))$ may be available in closed-form, while gradients of the log-likelihood $\log p(\mathbf{y} \mid \mathbf{w}, \boldsymbol{\theta})$ are evaluated using Monte Carlo samples and the local reparameterization trick [13]. Following this procedure, we obtain an optimal set of parameters for $q_{\boldsymbol{\phi}}(\mathbf{w})$, with which we can sample posterior draws for the logit layer. Hereafter, we will refer to this optimal set as $\{\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\phi}}\}$.

4.3 Assuming Gaussian Variational Posteriors

In this work, we choose the diagonal Gaussian as the family for $q_{\widehat{\phi}}(\mathbf{w})$. We write

$$q_{\widehat{\phi}}(\mathbf{w}) = \mathcal{N}(\mathbf{m}, \text{diag}(\mathbf{v})), \quad \widehat{\phi} = \{\mathbf{m}, \mathbf{v}\}, \tag{5}$$

with mean vector $\mathbf{m}$ and a covariance matrix with diagonal elements $\mathbf{v}$. This assumes that the variational posterior fully factorizes over the elements of $\mathbf{w}$. An advantage of this choice is that it ensures that the predicted logits $\boldsymbol{f}$ will follow a multivariate Gaussian as well. Note, however, that RATE would still be applicable to more complex choices of variational families. Using Equations (3) and (5), we may then derive the implied distribution over the logits as

$$\boldsymbol{f} \,|\, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\mathbf{H}(\widehat{\boldsymbol{\theta}})\mathbf{m} + \mathbf{b}, \mathbf{H}(\widehat{\boldsymbol{\theta}})\text{diag}(\mathbf{v})\mathbf{H}(\widehat{\boldsymbol{\theta}})^\intercal). \tag{6}$$

While the elements of $\mathbf{w}$ are independent, dependencies in the input data (via the activations $\mathbf{H}(\widehat{\boldsymbol{\theta}})$) have induced non-diagonal covariance between the elements of $\boldsymbol{f}$. The resulting distribution in Eq. (6) can then be used to calculate effect size analogues and RATE.

4.4 Covariance Projection Operator

After having conducted (variational) Bayesian inference, we use posterior draws from Eq. (6) to define an effect size analogue for neural networks. We could use the Moore-Penrose pseudo-inverse as proposed in [1] but, in the case of highly correlated inputs, this operator suffers from instability (see a small simulation study in Appendix 8), explaining the well-known phenomenon of linear regression suffering in the presence of collinearity. While regularization poses a viable solution to this problem, the selection of an optimal penalty parameter is not always a straightforward task. As a result, we propose a much simpler projection operator that will prove to be very effective, particularly in application areas where data measurements can be perfectly collinear (e.g. pixels in an image). Our solution is to use a linear measure of dependence separately for each predictor based on the sample covariance. Namely, for each of the p input variables

$$\widetilde{\boldsymbol{\beta}} := \text{cov}(\mathbf{X}, \boldsymbol{f}) = [\text{cov}(\mathbf{x}_1, \boldsymbol{f}), \ldots, \text{cov}(\mathbf{x}_p, \boldsymbol{f})]. \tag{7}$$

Since it is based on the sample covariance, the effect size analogue has the form $\widetilde{\boldsymbol{\beta}} = \mathbf{X}^\intercal\mathbf{C}\boldsymbol{f}/(n-1)$ — where $\mathbf{C} = \mathbf{I} - \mathbf{1}\mathbf{1}^\intercal/n$ denotes the centering matrix, $\mathbf{I}$ is an n-dimensional identity matrix and $\mathbf{1}$ is an n-dimensional vector of ones. Probabilistically, since we assume the posterior of the logits to be normally distributed, the above is equivalent to assuming that $\widetilde{\boldsymbol{\beta}} \,|\, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Omega})$ where

$$\boldsymbol{\mu} = \frac{1}{n-1}\mathbf{X}^\intercal\mathbf{C}\mathbf{H}(\widehat{\boldsymbol{\theta}})\mathbf{m} \quad \text{and} \quad \boldsymbol{\Omega} = \frac{1}{(n-1)^2}\mathbf{X}^\intercal\mathbf{C}\mathbf{H}(\widehat{\boldsymbol{\theta}})\text{diag}(\mathbf{v})\mathbf{H}(\widehat{\boldsymbol{\theta}})^\intercal\mathbf{C}^\intercal\mathbf{X}. \tag{8}$$

Intuitively, each element in $\widetilde{\boldsymbol{\beta}}$ represents some measure of how well the original data at the input layer explains the variation between observation classes. Moreover, under this approach, if two predictors $\mathbf{x}_r$ and $\mathbf{x}_s$ are almost perfectly collinear, then the corresponding effect sizes will also be very similar since $\text{cov}(\mathbf{x}_r, \boldsymbol{f}) \approx \text{cov}(\mathbf{x}_s, \boldsymbol{f})$. To build a better intuition for identifiability under this covariance projection, recall simple linear regression where ordinary least squares (OLS) estimates are unique modulo the span of the data [14]. A slightly different issue will arise for the effect size analogues computed via Eq. (7), where now two estimates are unique modulo the span of a vector of ones, or $span\{\mathbf{1}\}$. We now make the following formal statement.

Claim 4.1. *Two effect size analogues computed via the covariance projection operators, $\widetilde{\boldsymbol{\beta}}_1 = cov(\mathbf{X}, \boldsymbol{f}_1)$ and $\widetilde{\boldsymbol{\beta}}_2 = cov(\mathbf{X}, \boldsymbol{f}_2)$, are equivalent if and only if the corresponding functions are related by $\boldsymbol{f}_1 = \boldsymbol{f}_2 + c\mathbf{1}$, where $\mathbf{1}$ is a vector of ones and c is some arbitrary constant.*

The proof of this claim is trivial and follows directly from the covariance being invariant with respect to changes in location. Other proofs connecting this effect size to classic statistical measures can be found in Appendix 9.

4.5 Closed-Form Centrality Measures and groupRATE Extensions

Under our modelling assumptions, the posterior distribution of $\widetilde{\boldsymbol{\beta}}$ is multivariate normal with an empirical mean vector $\boldsymbol{\mu}$ and positive semi-definite covariance/precision matrix $\boldsymbol{\Omega} = \boldsymbol{\Lambda}^{-1}$. Given

these values, we may partition such that, for the j-th input variable, $\boldsymbol{\mu} = (\mu_j; \boldsymbol{\mu}_{-j})$ and

$$\boldsymbol{\Omega} = \begin{pmatrix} \omega_j & \boldsymbol{\omega}_{-j}^\intercal \\ \boldsymbol{\omega}_{-j} & \boldsymbol{\Omega}_{-j} \end{pmatrix}, \quad \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_j & \boldsymbol{\lambda}_{-j}^\intercal \\ \boldsymbol{\lambda}_{-j} & \boldsymbol{\Lambda}_{-j} \end{pmatrix}.$$

Now we may compute RATE values using Eq. (2), which in the case of Gaussian distributions, has the following closed form [1]

$$\text{KLD}(\widetilde{\beta}_j) = \frac{1}{2}\left[\text{tr}(\boldsymbol{\Omega}_{-j}\boldsymbol{\Lambda}_{-j}) - \log|\boldsymbol{\Omega}_{-j}\boldsymbol{\Lambda}_{-j}| - (p-1) + \delta_j(\widetilde{\beta}_j - \mu_j)^2\right] \tag{9}$$

where $\delta_j = \boldsymbol{\lambda}_{-j}^\intercal \boldsymbol{\Lambda}_{-j}^{-1}\boldsymbol{\lambda}_{-j}$ and characterizes the implied linear rate of change of information when the effect of any predictor is absent — thus, providing a natural (non-negative) numerical summary of the role of each $\widetilde{\beta}_j$ in the multivariate distribution.

Depending on the application setting, one might be interested in assessing the joint importance of multiple inputs. For example, assume that we have prior knowledge about how sets of variables are related, and we are interested in ranking these groups rather than individual predictors. The closed-form RATE criterion in Eq. (9) can be extended to also assess the centrality of a group J (where J is a set of indices) via the following

$$\text{KLD}(\widetilde{\boldsymbol{\beta}}_J) = \frac{1}{2}\left[\text{tr}(\boldsymbol{\Omega}_{-J}\boldsymbol{\Lambda}_{-J}) - \log|\boldsymbol{\Omega}_{-J}\boldsymbol{\Lambda}_{-J}| - (p-m) + (\widetilde{\boldsymbol{\beta}}_J - \boldsymbol{\mu}_J)^\intercal \boldsymbol{\Delta}_J(\widetilde{\boldsymbol{\beta}}_J - \boldsymbol{\mu}_J)\right] \tag{10}$$

where m is used to denote the number of input variables that belong to group J, and $\widetilde{\boldsymbol{\beta}}_J$ is used to indicate the elements of the vector of $\widetilde{\boldsymbol{\beta}}$ corresponding to just those m inputs. Using Eq. (10), we define $\text{groupRATE}(\widetilde{\boldsymbol{\beta}}_J) := \text{KLD}(\widetilde{\boldsymbol{\beta}}_J)/\sum_L \text{KLD}(\widetilde{\boldsymbol{\beta}}_L)$. Note that this ability to determine both individual and group-wise importance for predictors is unique to our interpretable BNN specification (in contrast to conventional mimic model approaches). Scalability for computing this association measure can be found in Appendix 11.

5 Results

In this section, we illustrate the performance of our interpretable BNN framework via simulation studies, as well as on a computer vision task and a problem within statistical genetics. Additional results (including applications to natural language processing and public policy relevance), details on utilized BNN architectures/training procedures, and mimic models can be found in Section 13.

5.1 Simulation Studies

A BNN similar to Section 4.1 was trained on simulated binary classification datasets with $n = \{10^3, 10^4, 10^5\}$ observations and $p = \{100, 300, 1000\}$ variables. Using the `make_classification` function from scikit-learn [15], clusters of predictors with complex multivariate interactions were generated. Class labels were generated such that only 10% of variables were truly associated [16]. For each simulated dataset, a BNN was trained and then importance measures for the input variables were calculated using one of four methods: (i) RATE computed on the test set, (ii-iii) the Gini importance score based on a random forest (RF) or a gradient boosting machine (GBM) mimic model trained on the training samples of the BNN, and (iv) the correlation between each input predictor and the labels in the test set. The mimic models were selected using random search cross-validation (details in Appendix 14) where possible, but for $n = 10^5$ the running time was prohibitively long and so default scikit-learn models were used. We then assessed the power of each method to identify truly associated variables. Results are based on 10 simulated replicates for each (n, p) combination. Figure 2 displays ROC curves for a subset of these scenarios. The remaining results are included in Figure 6 and accompanied by runtimes in Table 2.

Overall, our results show that RATE at least matches (and often outperforms) the RF mimic models. Another key advantage of RATE is that it has no tuning parameters. Furthermore, results for the simple pixel correlation highlight that RATE can detect more than the linear effects learned by the BNN. Furthermore, these results show that the cross-validation (CV) of the mimic models is essential for correct variable importances. This is computationally costly, and we found empirically that RATE is faster than RF with CV (even for parallelized CV and RATE; see Table 2). These empirical timings indicate that the running time of RATE scales better with n than with p (compared to the RF mimic with CV).

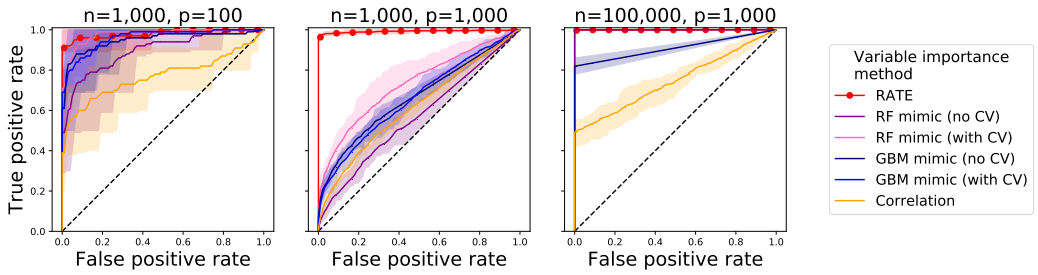

Figure 2: ROC curves evaluating the ability of each variable importance method to identify causal variables in the simulation study. The solid curves and shaded areas depict the mean ROC $\pm$ and the 10^{th}, 90^{th} quantiles, respectively. The AUCs are available in Table 1.

5.2 Image Classification using MNIST

We construct binary classification tasks using digits from MNIST [17]. Datasets consisted of $n \approx 12,000$ images and $p = 324$ pixels (after cropping). Results in the main text are based on comparing (i) ones and zeros and (ii) ones and eights. The BNN used in this analysis contained a single convolutional layer, followed by two fully-connected layers — satisfying the architectural requirements outlined in Section 4.1. Multi-class extensions are in Appendix 15.

Following training, we compute the RATE values for each pixel, where a high value indicates that a pixel is important for the network when differentiating between the two classes, as shown in Fig. 3A and B. Note that RATE values do not provide information about the class-specific associations, but we can assign a class to each pixel using the sign of $\mathbb{E}[p(\widetilde{\beta}_j \mid \mathbf{X}, \mathbf{y})]$. The pixels identified by RATE are consistent with human intuition. When distinguishing between zero and one, the most important pixels are in the center (where the vertical line of a one would appear) and in a ring (corresponding to the shape of zero). Similarly, for ones and eights, the shape of an eight is clearly visible.

While our results show a plausible set of important pixels under visual inspection, a natural followup analysis is to assess if these pixels are important for the network when it makes an out-of-sample prediction. We calculate prediction accuracy as certain pixels in the test images are shuffled, thus de-correlating the observations and their labels. Figure 3C shows the test set accuracy as progressively larger subsets of pixels are shuffled, chosen according to the rank of their RATE values, their rank according to various mimic models (linear, RF, GBM), or completely at random. The test set accuracy decreases much more steeply when pixels with the highest RATE or RF mimic (with CV) values are shuffled versus when pixels are selected using other approaches. This indicates that our method works almost as well as the state-of-the-art under this metric. Interestingly, our method yields more visually appealing results, as shown in Fig. 7. Most importantly, our method is significantly better than the RF mimic (no CV) which uses the defaults in scikit-learn; the running time of cross-validation for RF is an order of magnitude larger than that of RATE (which requires no tuning), even when the cross-validation is performed over 10 cores (see Table 3).

5.3 Gene Scores for Heterogeneous Stock of Mice Genetic Study

Lastly, we analyze median B220 content in a heterogeneous stock of mice dataset [18] from the Wellcome Trust Centre for Human Genetics. While we have mostly focused on classification, our method is readily applicable to the regression setting as well. This dataset contains $n = 1814$ individuals genotyped for $p = 10346$ single nucleotide polymorphisms (SNPs) with minor allele frequencies above 5% (for details, see Appendix 12). Traditionally, variable selection for individually associated SNPs has been regarded as a failure when applied to complex traits [19, 20]. As a result, recent approaches have aimed to combine SNPs within a chromosomal region to detect more biologically relevant genes and enriched pathways [21, 22]. While mimic models cannot be applied to this task, our interpretable BNN framework can be used to identify important groups of input variables using groupRATE (Eq. (10)). Here, we use the Mouse Genome Database (MGD) [23], and group together SNPs with genomic positions that fall within the regulatory regions of the same genes.

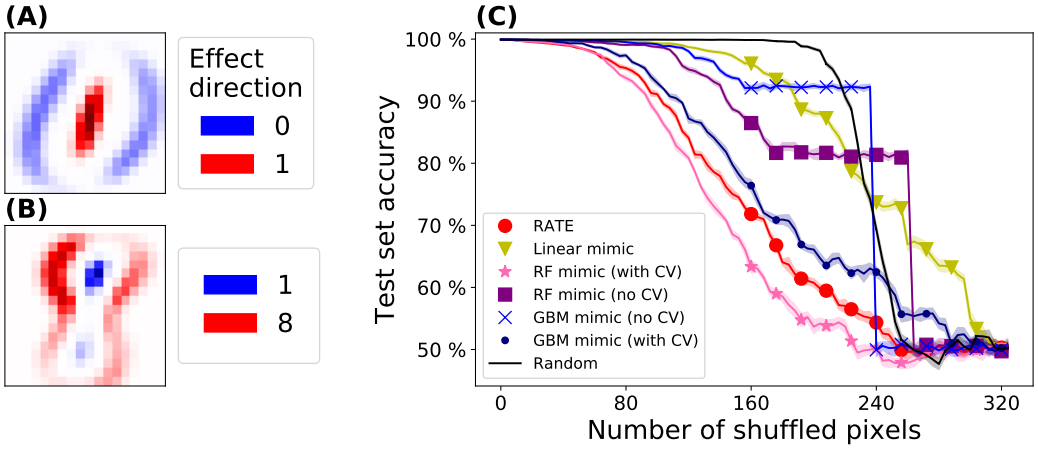

Figure 3: RATE values associated with each pixel in binary classification tasks using (A) zeros and ones and (B) ones and eights from the MNIST dataset. Colors indicate the class favored by a pixel, as determined by the sign of the posterior mean for the corresponding effect size analogue. Panel (C) shows the test set classification accuracies when pixels are shuffled according to their RATE values, mimic models, or at random. The solid lines and shaded areas are the mean test accuracy $\pm$ one standard deviation over ten repeated shuffles. RATE is outperformed by the RF mimic model, but has a shorter empirical run time due to RF cross-validation (see Table 3).

This resulted in 5121 total genes (or groups of variables) across the 19 chromosomes in the mouse genome. After having trained our BNN, we run groupRATE on each of these groups to create a gene association score. These scores are illustrated as a Manhattan plot in Figure 4. Here, we are able to detect many genomic regions known to be associated with the immunological traits.

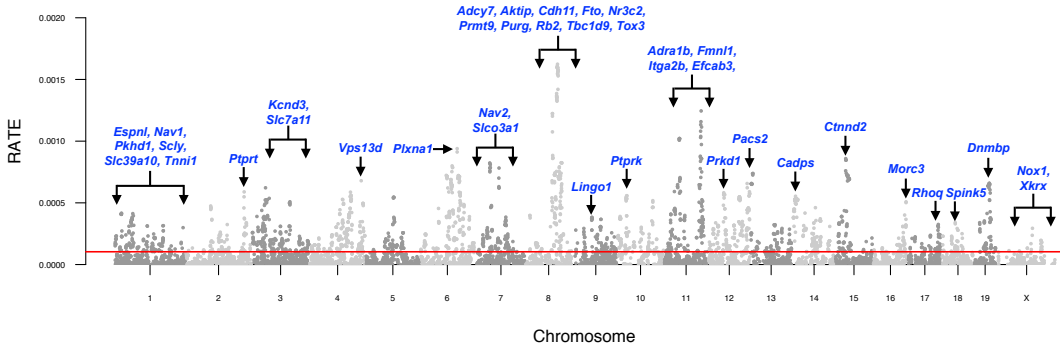

Figure 4: The groupRATE results for genes in the heterogeneous stock of mice genome-wide association study (GWAS) from the Wellcome Trust Centre for Human Genetics. SNP-to-gene annotations and genomic positions were determined using the Mouse Genome Database (MGD).

6 Discussion

We developed a novel global interpretability method for deep neural networks. We focused on settings in which predictor variables are intrinsically meaningful, and it is of scientific relevance to rank these predictor variables. We worked in a very flexible variational Bayes approach to deep learning, proposed a sample covariance operator as our effect size analogue, and extended the recently proposed RelATive cEntrality (RATE) measure to our setting, providing closed-form solutions for its implementation, and proposing groupRATE, an extension to rank groups of variables. Lastly, we illustrated the performance of our framework in broad applications including computer vision, statistical genetics, natural language processing, and public policy. Our method outperforms or

achieves performance on par with the state-of-the-art, while avoiding the need for a separate, time consuming tuning step.

7 Software Availability

Software for implementing the interpretable Bayesian DNN framework with RATE significance measures is carried out in R and Python code, which is available at `https://bit.ly/2JZSALV`.

References

[1] L. Crawford *et al.*, "Variable prioritization in nonlinear black box methods: A genetic association case study," *Annals of Applied Statistics*, 2019.

[2] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv:1702.08608*, 2017.

[3] D. Erhan *et al.*, "Visualizing higher-layer features of a deep network," *University of Montreal*, vol. 1341, no. 3, p. 1, 2009.

[4] N. Frosst and G. Hinton, "Distilling a neural network into a soft decision tree," *arXiv preprint arXiv:1711.09784*, 2017.

[5] F. Wang and C. Rudin, "Falling rule lists," in *Artificial Intelligence and Statistics*, 2015, pp. 1013–1022.

[6] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[7] X. Chen *et al.*, "A forest-based approach to identifying gene and gene–gene interactions," *Proceedings of the National Academy of Sciences*, vol. 104, no. 49, pp. 19 199–19 203, 2007.

[8] Z. Che *et al.*, "Interpretable deep models for icu outcome prediction," in *AMIA Annual Symposium Proceedings*, vol. 2016. American Medical Informatics Association, 2016, p. 371.

[9] L. Crawford *et al.*, "Bayesian approximate kernel regression with variable selection," *Journal of the American Statistical Association*, vol. 113, no. 524, pp. 1710–1721, 2018.

[10] G. E. Hinton and D. Van Camp, "Keeping neural networks simple by minimizing the description length of the weights," in *Proceedings of the Sixth Annual Conference on Computational Learning Theory*. ACM, 1993, pp. 5–13.

[11] D. Barber and C. M. Bishop, "Ensemble learning in Bayesian neural networks," *NATO ASI Series F Computer and Systems Sciences*, vol. 168, pp. 215–238, 1998.

[12] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 2348–2356.

[13] D. P. Kingma *et al.*, "Variational dropout and the local reparameterization trick," in *Advances in Neural Information Processing Systems*, 2015, pp. 2575–2583.

[14] S. Wold *et al.*, "The collinearity problem in linear regression. the partial least squares (pls) approach to generalized inverses," *SIAM Journal on Scientific and Statistical Computing*, vol. 5, no. 3, pp. 735–743, 1984.

[15] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] I. Guyon *et al.*, "Competitive baseline methods set new standards for the nips 2003 feature selection benchmark," *Pattern recognition letters*, vol. 28, no. 12, pp. 1438–1444, 2007.

[17] Y. LeCun, "The MNIST Database of Handwritten Digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[18] W. Valdar *et al.*, "Genome-wide genetic association of complex traits in heterogeneous stock mice," *Nature Genetics*, vol. 38, no. 8, pp. 879–887, 2006. [Online]. Available: http://dx.doi.org/10.1038/ng1840

[19] T. A. Manolio *et al.*, "Finding the missing heritability of complex diseases," *Nature*, vol. 461, no. 7265, pp. 747–753, 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/19812666

[20] P. M. Visscher *et al.*, "Five years of gwas discovery," *American Journal of Human Genetics*, vol. 90, no. 1, pp. 7–24, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0002929711005337

[21] X. Zhu and M. Stephens, "Large-scale genome-wide enrichment analyses identify new trait-associated genes and pathways across 31 human phenotypes," *Nature Communications*, vol. 9, no. 1, p. 4361, 2018.

[22] W. Cheng *et al.*, "Epsilon-genic effects bridge the gap between polygenic and omnigenic complex traits," *bioRxiv*, p. 597484, 2019. [Online]. Available: http://biorxiv.org/content/early/2019/04/02/597484.abstract

[23] J. A. Blake *et al.*, "Mgd: the mouse genome database," *Nucleic Acids Research*, vol. 31, no. 1, pp. 193–195, 2003. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/12519980

[24] A. L. Maas *et al.*, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*. Association for Computational Linguistics, 2011, pp. 142–150.

[25] J. Angwin *et al.*, "Machine bias: There's software used across the country to predict future criminals. And it's biased against blacks," *ProPublica*, 2016.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] R. F. Baumeister *et al.*, "Bad is Stronger than Good." *Review of General Psychology*, vol. 5, no. 4, p. 323, 2001.

[28] J. Dressel and H. Farid, "The accuracy, fairness, and limits of predicting recidivism," *Science Advances*, vol. 4, no. 1, p. eaao5580, 2018. [Online]. Available: http://advances.sciencemag.org/content/4/1/eaao5580.abstract

Supplementary Material

8 Robustness of the Covariance Projection Operator in the Presence of Collinearity

In this section, our goal is to motivate the use of the covariance projection operator for the effect size analogue in Bayesian neural networks. We do this via a small simulation study which shows that the conventional linear estimation of regression coefficients is unstable in applications with highly collinear predictors. Here, we generate a synthetic design matrix with $n = 5000$ individuals and $p = 2$ covariates ($\mathbf{x}_1$ and $\mathbf{x}_2$) randomly drawn from standard normal distributions. We then assess two simulation scenarios with continuous outcomes created under the following linear model

$$\mathbf{y} = 2\mathbf{x}_1 - 2\mathbf{x}_2 + \boldsymbol{\varepsilon}, \quad \boldsymbol{\beta} = [2, -2], \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

In the first simulation scenario, $\mathbf{x}_1$ and $\mathbf{x}_2$ are uncorrelated; while, in the second scenario, the two covariates are set to share a Pearson correlation coefficient of $\rho = 0.999$. In each case, we compare the classic ordinary least squares (OLS) estimate for regression coefficients $\widehat{\boldsymbol{\beta}} = (\mathbf{X}^\intercal\mathbf{X})^{-1}\mathbf{X}^\intercal\mathbf{y}$ and the proposed covariance effect size analogue $\widetilde{\boldsymbol{\beta}} = [\text{cov}(\mathbf{x}_1, \mathbf{y}), \text{cov}(\mathbf{x}_2, \mathbf{y})]$. Figure 5 depicts the results for both cases repeated 100 different times. In Figure 5A, we see that both types of estimators are able to properly capture the true effects when the predictors are uncorrelated. This finding is expected. However, in the extremely collinear scenario with $\mathbf{x}_1 \approx \mathbf{x}_2$, the total true effect size in the simulation is effectively equal to $\beta = 2 - 2 = 0$. The OLS estimators are unstable under this condition, while the covariance effect size analogues accurately and robustly estimate this value (see Figure 5B).

9 Connection between the Covariance Projection and Marginal Association Tests

In this section, we prove a connection between the covariance projection operator and the conventional hypothesis testing strategies for marginal feature associations. Assume that we have an n-dimensional outcome variable $\mathbf{y}$ that is to be modeled by an $n \times p$ design matrix $\mathbf{X}$. In linear regression, a simple (yet effective) approach is to take each covariate $\mathbf{x}_j$ in turn and assess associations based upon a two-tailed alternative hypothesis. The significance of this test is then summarized via p-values (e.g. $\widehat{p}_j$ for feature j), which may then be ranked in the order of importance from smallest to largest. Here, we show that the effect size analogues $\widetilde{\boldsymbol{\beta}}$ correspond exactly to the test statistics for this frequented univariate approach.

Begin by recalling that the covariance projection operator simply produces the sample covariance between a given predictor variable $\mathbf{x}_j$ and the model predictions $\boldsymbol{f}$ — where both largely positive or negative covariances are informative. Next, recall that the sample covariance between two random variables is equal to their Pearson correlation coefficient (ρ) multiplied by their respective standard errors σ_X and σ_Y,

$$\text{cov}(X, Y) = \rho\, \sigma_X \sigma_Y. \tag{11}$$

The standard formula for p-values starts by calculating a t-statistic of the following form

$$T_j = \rho_j \sqrt{\frac{n-2}{1-\rho_j^2}}, \quad j = 1, \ldots, p. \tag{12}$$

Corresponding p-values are then computed by comparing these values to a Student's t-distribution function under the null hypothesis — with the intuition being that larger test statistics will result in smaller p-values.

We now verify that these transformations are all monotonic — thus, our proposed covariance effect size analogue will result in the same ranking of variable importance as the classical t-test.

Theorem 1. *If two predictor variables have covariance effect size analogues such that* $\widetilde{\beta}_1 = cov(\mathbf{x}_1, \boldsymbol{f}) > cov(\mathbf{x}_2, \boldsymbol{f}) = \widetilde{\beta}_2$*, then the resulting p-values from a t-test with these features will have the relationship* $\widehat{p}_1 < \widehat{p}_2$.

Proof. Consider the covariance projection operation on two different predictor variables, $\text{cov}(\mathbf{x}_1, \boldsymbol{f}) > \text{cov}(\mathbf{x}_1, \boldsymbol{f})$. Since standard deviations are positive

$$\text{cov}(\mathbf{x}_1, \boldsymbol{f})\sigma_{\mathbf{x}_1}\sigma_{\boldsymbol{f}} > \text{cov}(\mathbf{x}_1, \boldsymbol{f})\sigma_{\mathbf{x}_2}\sigma_{\boldsymbol{f}} \quad \iff \quad \rho_1 > \rho_2.$$

The same applies when multiplying both sides by $\sqrt{n-2}$. Also note that since we are concerned with the magnitude of covariances (and subsequently correlations),

$$\rho_1 > \rho_2 \quad \iff \quad \sqrt{1-\rho_1} \leq \sqrt{1-\rho_2}.$$

Therefore we conclude that

$$\rho_1\sqrt{\frac{n-2}{1-\rho_1^2}} > \rho_2\sqrt{\frac{n-2}{1-\rho_2^2}} \quad \iff \quad T_1 > T_2.$$

Since the distribution function is monotonic, $\widehat{p}_1 < \widehat{p}_2$. □

10 Connection between RATE and Mutual Information

Given an effect size analogue $\widetilde{\boldsymbol{\beta}}$, the RATE criterion for any variable j is defined as the normalized Kullback-Leibler divergence between (i) the conditional posterior predictive distribution $p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\beta}_j = 0)$ with the effect of the j-th predictor being set to zero, and (ii) the marginal distribution $p(\widetilde{\boldsymbol{\beta}}_{-j})$ with the effects of the j-th predictor being integrated out:

$$\text{RATE}(\widetilde{\beta}_j) = \frac{\text{KL}\left(p(\widetilde{\boldsymbol{\beta}}_{-j}) \,\|\, p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\beta}_j = 0)\right)}{\sum_\ell \text{KL}\left(p(\widetilde{\boldsymbol{\beta}}_{-\ell}) \,\|\, p(\widetilde{\boldsymbol{\beta}}_{-\ell} \mid \widetilde{\beta}_\ell = 0)\right)}\,.$$

In this section, we compare the Kullback-Leibler divergence specified on the numerator of the RATE criterion to the mutual information (MI) between $\widetilde{\boldsymbol{\beta}}_{-j}$ and $\widetilde{\boldsymbol{\beta}}_j$. Notice by simplifying the definition

$$\begin{aligned}
\text{MI}(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j) &= \iint p(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j) \log \frac{p(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j)}{p(\widetilde{\boldsymbol{\beta}}_{-j})p(\widetilde{\boldsymbol{\beta}}_j)} d\widetilde{\boldsymbol{\beta}}_{-j} d\widetilde{\boldsymbol{\beta}}_j && (13)\\
&= \iint p(\widetilde{\boldsymbol{\beta}}_j)p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\boldsymbol{\beta}}_j) \log \frac{p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\boldsymbol{\beta}}_j)}{p(\widetilde{\boldsymbol{\beta}}_{-j})} d\widetilde{\boldsymbol{\beta}}_{-j} d\widetilde{\boldsymbol{\beta}}_j && (14)\\
&= \int p(\widetilde{\boldsymbol{\beta}}_j) \text{KL}\left(p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\beta}_j) \,\|\, p(\widetilde{\boldsymbol{\beta}}_{-j})\right) d\widetilde{\boldsymbol{\beta}}_j\,. && (15)
\end{aligned}$$

While the RATE criterion compares the marginal distribution $p(\widetilde{\boldsymbol{\beta}}_{-j})$ to the conditional distribution $p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\beta}_j = 0)$ with the effect of the j-th predictor being set to zero, the mutual information criterion compares $p(\widetilde{\boldsymbol{\beta}}_{-j})$ to the conditional distribution $p(\widetilde{\boldsymbol{\beta}}_{-j} \mid \widetilde{\beta}_j)$ for all the possible values of $\widetilde{\beta}_j$.

Whenever the effect size analogue $\widetilde{\boldsymbol{\beta}}$ follows a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Omega})$, the RATE criterion for the j-th variable is given by Equation (9) in the main text. In the same setting, the mutual information $MI(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j)$ can also be computed analytically as follows:

$$MI(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j) = \frac{1}{2}\log\left(\alpha_j |\boldsymbol{\Omega}_{-j}||\boldsymbol{\Omega}|^{-1}\right) \tag{16}$$

where, for simplicity, we use the same notations as in Section 4.5. Importantly, note that $\text{MI}(\widetilde{\boldsymbol{\beta}}_{-j}, \widetilde{\boldsymbol{\beta}}_j)$ only depends on the values of the covariance/precision matrix $\boldsymbol{\Omega} = \boldsymbol{\Lambda}^{-1}$. This is in contrast to the RATE criterion which also depends on an estimate of the posterior mean $\boldsymbol{\mu}_j$. Additionally, the mutual information criterion is equal to 0 if and only if $\widetilde{\boldsymbol{\beta}}_{-j}$ and $\widetilde{\boldsymbol{\beta}}_j$ are independent.

11 Brief Note on Scalability

The scalability of the full RATE calculation (including the effect size analogue, posterior mean, and posterior covariance) is $\mathcal{O}(pn^2 + p^2n + p^4)$ for n observations and p variables. Hence, the leading order term is $\mathcal{O}(p^4)$ which is driven by p independent $\mathcal{O}(p^3)$ operations (i.e. the matrix inversions in Equation (9) in the main text). Unfortunately, this restricts RATE to datasets of size $n \lessapprox 10^5$ and $p \lessapprox 10^4$. Interestingly, we empirically found that run times for RATE are less than the mimic models when the time required for cross-validation is also included (see Tables 2 and 3).

12 Experimental Datasets

We use four real datasets in the present study. The first comes from the MNIST database of handwritten images (`http://yann.lecun.com/exdb/mnist/`) [17]. The downloaded digits had already been size-normalized and centered in a fixed-size image with 28×28 dimensions. It has been noted that the error rate of classification methods can improve when the digits are centered by bounding box rather than center of mass. To this end, we further cropped the images with a 5-pixel border — resulting in a final dataset with digits of size 18×18. We note that this border region contained only zeros in the vast majority of the images, and so the pixels in these regions were not informative. The binary classification task involving zeros and ones had training and test set sizes of 12,665 and 2115 pixels respectively.

The heterogeneous stock of mice consists of 1,904 individuals from 85 families, all descended from eight inbred progenitor strains sequenced by the Wellcome Trust Centre for Human Genetics (`http://mtweb.cs.ucl.ac.uk/mus/www/mouse/index.shtml`) [18]. The data contains 129 quantitative traits that are classified into 6 broad categories including behavior, diabetes, asthma, immunology, haematology, and biochemistry. All phenotypes were previously corrected for sex, age, body weight, season, and year effects. A total of 12,226 autosomal SNPs were available for all mice. For individuals with missing genotypes, we imputed missing values by the mean genotype of that SNP in their family. All polymorphic SNPs with minor allele frequency above 5% in the training data were used. In the heterogenous stock of mice data applications, SNPs are mapped to the closest neighboring gene(s) using the Mouse Genome Database (`http://www.informatics.jax.org`).

The Large Movie Review dataset included the 1,000 most frequently used words (excluding padding, unknown, and start characters) for 50,0000 reviews (`http://ai.stanford.edu/~amaas/data/sentiment/`) [24]. These reviews were split into equally sized test and training sets. The unformatted data consists of sequences of integers, where each integer corresponds to a word. These were encoded using bag-of-words, which resulted in each review being represented by a 1000-dimensional vector whose j-th element denotes the relative frequency of the j-th most frequent word in the corpus. The relative frequency is the number of times a word appears in a document divided its total number of appearances in the training or test set.

For the COMPAS analysis, we downloaded the same dataset that *ProPublica* used on criminal defendants from Broward County, Florida and follow the recommended data cleaning procedures (`https://github.com/propublica/compas-analysis`) [25]. Individuals with charge dates more than thirty days before or after arrest were dropped because the arrest was likely associated with a different crime than the one inciting a COMPAS score. The dataset is also pruned for individuals who either recidivated in two years or have two years outside a correctional facility. We also pruned for people whose COMPAS-scored crime was not an ordinary traffic offense. Like *ProPublica*, we binned the response into low risk versus medium and high risk (the multinomial case is also reviewed later in the Supplemental Material), but deviate from *ProPublica* by omitting the "two-year recidivism" covariate from the considered feature set.

13 Architecture and Training Procedure for Bayesian Neural Networks

We now detail the Bayesian neural network architectures and training procedures used to derive the results presented in the main text:

- In the simulation study detailed in Section 5.1, the network consisted of two hidden, fully-connected layers with 512 units each and ReLU activations. The output layer is specified as a hierarchical Bayesian model, as described in Section 4.1, with sigmoid activation.
- For the MNIST study in section 5.2, the network had a convolutional layer with 32 filters and stride 5, whose flattened output was passed to a fully-connected layer with 512 units and ReLU activation. The final layer was again specified with Bayesian hierarchical priors with a sigmoid activation.
- For the heterogenous stock of mice dataset in Section 5.3, the BNN had the same architecture as the simulation studies, but batch normalizations were used with each of the hidden layers. The final layer had an identity activation since phenotype of interest was continuous and required a regression-based task.

In all three cases, the BNNs were trained for 20 epochs using early stopping (with a patience of 2 epochs) based on the accuracy of (i) a held-out validation set that contained 20% of the training examples (for the simulations and MNIST analysis), or (ii) the mean squared error on the training set (which was done for the mice genetic study since there was insufficient data for a distinct validation set). The Adam optimizer with a learning rate of 1e-3 was used in all three cases [26].

14 Cross-Validation of Mimic Models

Here, we used random search with 3-fold cross-validation for the random forest and gradient boosting machine mimic models. Each random search fit ten different models. This was done using scikit-learn [15]. The running time for the simulation studies described in Section 5.1 of the main text are shown in Table 2. The pixel-wise correlation is not included as its running time is negligible. If no cross-validation was performed the default scikit-learn model was used as the mimic.

15 Additional MNIST Results

Binary Classification Task

In addition to the results in the main text, we also generated visualizations of the variable importances according to linear, random forest, and gradient boosting machine mimic models. These are displayed for the binary classification of zeros and ones from MNIST in Figure 7. The second and third columns show the importance of cross-validation for these mimic models — since the mimics selected by cross-validation tend to produce more visually compelling results. Figure 3C supports this conclusion, as the test set accuracy decreases more quickly when the pixels are shuffled according a mimic model selected using cross-validation than for the corresponding default mimic model.

Empirical Running Times

The running times required to compute the variable importance on a binary classification task from MNIST, where $n = 12,665$ images and $p = 324$ pixels, are shown in Table 3.

Multi-Class Analysis

In the multi-class case, we set up the neural network to contain 10 output nodes (corresponding to the 10 digits of MNIST). We compute effect size analogues and RATE values for each node — meaning that each pixel has a set of 10 RATE values that indicate its importance in identifying each of the 10 digits. These RATE values are shown for each node in Figure 8, which illustrates how each feature associates with a particular digit. Results for nodes 0, 3, 6, 8 and 9 are particularly easy to understand visually.

In order to evaluate the pixel rankings produced by RATE, we perform two experiments. In the first, any pixel with a RATE value greater than $1/p$ for any of the 10 classes is shuffled (221 pixels in total). In the second experiment, the remaining pixels are shuffled — this is the set of pixels that are not important for distinguishing between digit classes according to RATE (103 pixels). In both experiments, the same number of randomly-selected pixels are selected and shuffled. The results for both experiments are shown in Figure 9. Figure 9A shows that removing high-RATE pixels causes the test accuracy to effectively resemble a random classifier (i.e. the red dotted line). Removing pixels at random at least retains some information. Figure 9B shows that the reverse to be true. After shuffling all the low-RATE pixels, the test accuracy is insignificantly reduces to 93.4% (i.e. reduction of 5.0%). However, shuffling the same number of pixels at random reduces the test accuracy to 68.6% (i.e. reduction of 29.8%).

16 Large Movie Review Sentiment Analysis

The Large Movie Review dataset [24] contains 50,000 reviews labeled as having either positive or negative sentiment. These are also split equally into test and training sets. The reviews are encoded using bag-of-words such that each observation is a D-dimensional vector for dictionary size D, with

the j-th element denoting the number of times that word j appears in the review. In this analysis, the dictionary consisted of the $D = 1000$ most frequent words in the entire corpus.

A Bayesian neural network with three fully-connected layers and ReLU activations was trained to predict sentiment from the encoded reviews. For comparison, we also trained a random forest (with hyperparameters selected using random grid search) and a logistic regression model. Random forest is a popular nonparametric, nonlinear model that has an established variable importance methodology based on Gini impurity, while classic logistic regression is conventionally interpretable as it provides odds ratios and associated p-values.

Figure 10 shows the top 15 most important words according to (A) a Bayesian DNN with RATE, (B) the random forest model and (C) logistic regression. For RATE, the directions of the effect (positive versus negative) are assigned via the sign of $\mathbb{E}[p(\widetilde{\beta}_j \mid \mathbf{X}, \mathbf{y})]$ for that word. The words identified by our approach are all associated with positive or negative sentiment, with 12 of the 15 most highly ranked words having negative sentiment. This reflects an established phenomenon from psychology which poses that negative sentiments tend to outweigh positive ones [27]. Furthermore, the results of our framework match many of the words that the other two methods deem as important. Out of the top 15 words ranked by RATE, 6 of them also appear in the equivalently ranked list for the random forest and 6 appear for logistic regression. For the 112 words with RATE values greater than the $1/p$ cutoff, 38 and 74 words appear in the equivalently ranked list for random forest and logistic regression, respectively. The Bayesian neural network utilizes rankings that make intuitive sense and produces results that are supported by established interpretable models.

17 COMPAS Risk Score Study

Binary Class Analysis

The criminal risk assessment tool, Correctional Offender Management Profiling for Alternative Sanctions (COMPAS), has been widely used to predict offender recidivism since its release in 1998. Recently, writers for *ProPublica* analyzed the algorithm on approximately 10000 individuals arrested in Broward County, Florida between 2013 and 2014 and found its predictions to be racially biased [25, 28]. Here, logistic regression revealed that being African American was significant in predicting COMPAS-assigned risk scores. Our aim in this section is to analyze this same data with a Bayesian DNN and RATE to examine the relationship between COMPAS-assigned risk scores and race.

To begin, we filtered the data to only include individuals who had either recidivated in two years, or had at least two years outside of a correctional facility. This resulted in a final dataset with 6172 observations with 5 covariate types: number of prior offenses, race (labeled as African American, Asian, Hispanic, Native American, or Other), gender (labeled as 1 for being female, 0 for otherwise), age group (labeled as older than 45 years old or less than 25), and the severity of charge. The COMPAS system classifies people into high, medium, and low risk categories. In the main text, we focus on the binary classification problem between the high and low risk categories. In the next subsection, we turn to the multi-class problem and jointly examine all three levels together.

In the case of the binary classification analysis, we follow the methodology described in Section 4. Here, we again fit a Bayesian neural network with three fully-connected layers and ReLU activations. We then estimated the variational distribution for the weights on the final layer. Next, we inferred the implied posterior distribution of the covariance effect size analogue and compute RATE measures for each covariate. The number of prior offenses was identified as the only significant variable in the model (see Figure 11A).

There is a difference between our results using a deep learning approach and *ProPublica*'s simple logistic model which identified racial factors (specifically an individual be identified as being African American) as being a large factor in predicting COMPAS risk scores. This motivated us to compare the distribution of prior offense counts across racial groups (see Figure 11B). Approximately two-thirds of the modeling sample is non-white, of whom 31.06% have no prior offenses and some individuals were recorded as having more than 30 previous accounts. In the white cohort only 39.04% were without prior offenses, but the tails of that distribution did not exceed far passed 15. A Kolmogorov-Smirnov (KS) test between the two groups yielded a p-value near zero — confirming that this statistic was inherently racially biased. This also explains why the DNN did not bother to

place any significant weight/prioritization on the other predictor variables that were included in the model.

To this end, we omitted the number of prior offenses from the neural network and subsequent analyses led to RATE identifying the African-American racial factor as being the most significant predictor in classifying COMPAS risk scores (see Figure 11C). The number of prior offenses essentially masked this effect. A DNN trained on the full dataset had a predictive accuracy of 75.5%. Fitting the same model with the number of prior offenses as its only input yielded an accuracy of 68.2% — but this is not a sizable improvement over the 67.49% predictive accuracy we observed when this variable was omitted.

Multi-Class Analysis

Instead of binning the COMPAS responses to create a binary classification problem, it is natural to consider the original three risk scores categories: low, medium, and high. Feeding the data matrix through a neural network to predict the multinomial case gives RATE values for each of the three possible class labels (see Figure 12A). This simply requires redefining $\sigma(\bullet)$ in Equation (3) to be the softmax function.

The number of prior offenses accounts for almost all of the relative significance for each class label, similar to the binary case. No other variable rises in importance to distinguish either the risk scores. Removing the number of prior offenses from the feature set and retraining the neural network gives RATE values for each of the three class labels (see Figure 12B) — which again mimic the result seen in the binary case.

Supplementary Figures

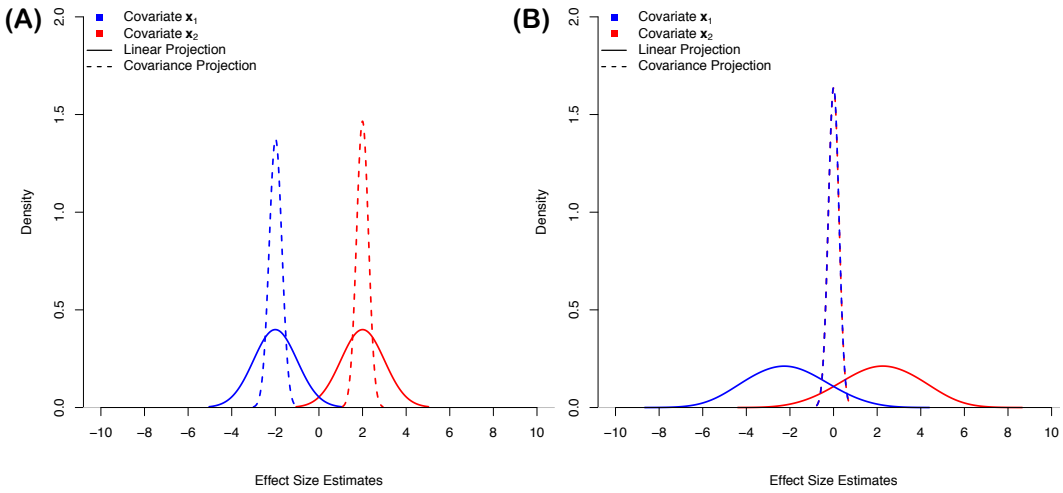

Figure 5: Results from a small simulation study showing the robustness of the covariance projection operator in the presence of collinear predictor variables. Synthetic data is generated as $\mathbf{y} = \mathbf{2x_1} - \mathbf{2x_2} + \varepsilon$ with $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. OLS effect sizes are compared as a baseline. In panel (A), outcomes variables are generated with uncorrelated predictors; while in panel (B), the two covariates have a Pearson correlation coefficient of $\rho = 0.999$.

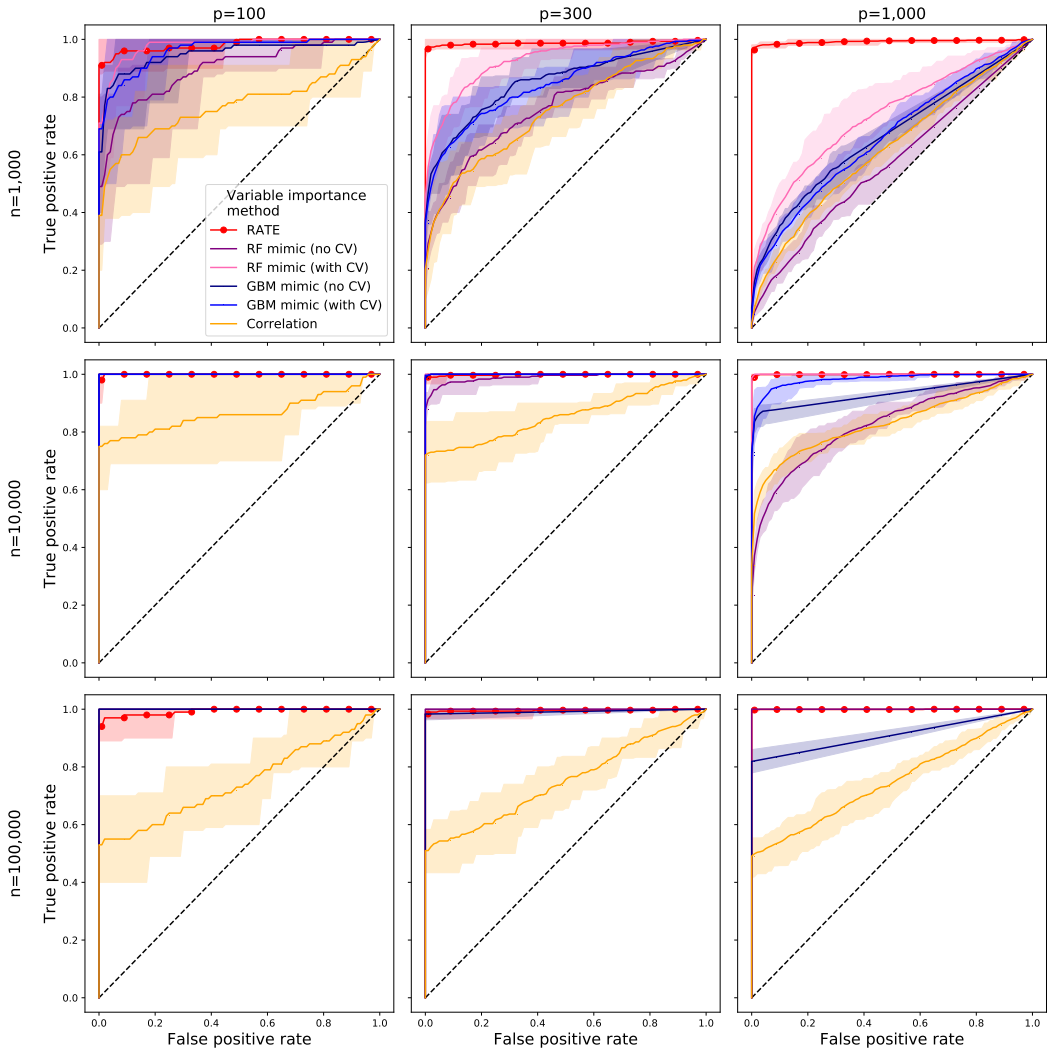

Figure 6: ROC curves evaluating the ability of each variable importance method to identify causal variables in the simulation study. For each combination of n samples and p predictors, ten datasets were simulated with 10% of variables being causal. The solid curves and shaded areas depict the mean ROC and 10^{th} and 90^{th} percentiles over the replicates, respectively. This information is also tabulated for the AUC in the legend.

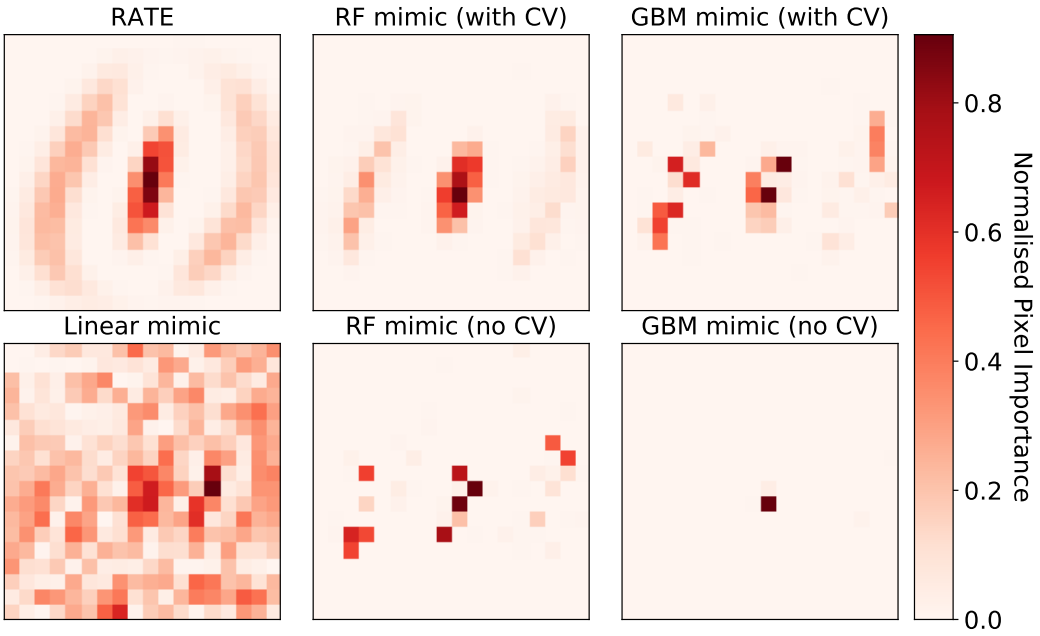

Figure 7: Visualizations of variable (pixel) importance computed using RATE and mimic models (logistic regression, random forest, and gradient boosting machine). The second and third columns show that cross-validation is required for the random forest and gradient boosting to produce pixel importances that correspond to human intuition for this dataset. The pixel importances for each method are normalized to sum to 1 across pixels, for comparison with RATE.

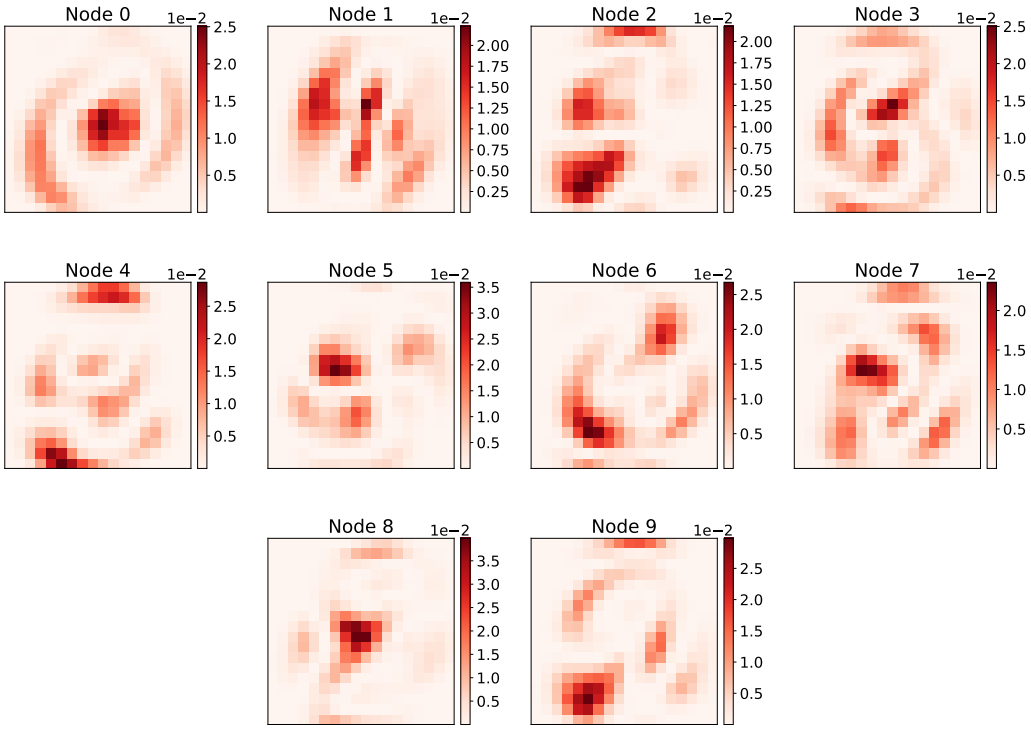

Figure 8: RATE values for a Bayesian neural network trained on the whole MNIST dataset. For the 10-class problem there are 10 output nodes and 10 corresponding RATE values for each pixel. The digit corresponding to each node can be seen clearly in several examples (e.g. 0, 3, 6, 8, and 9).

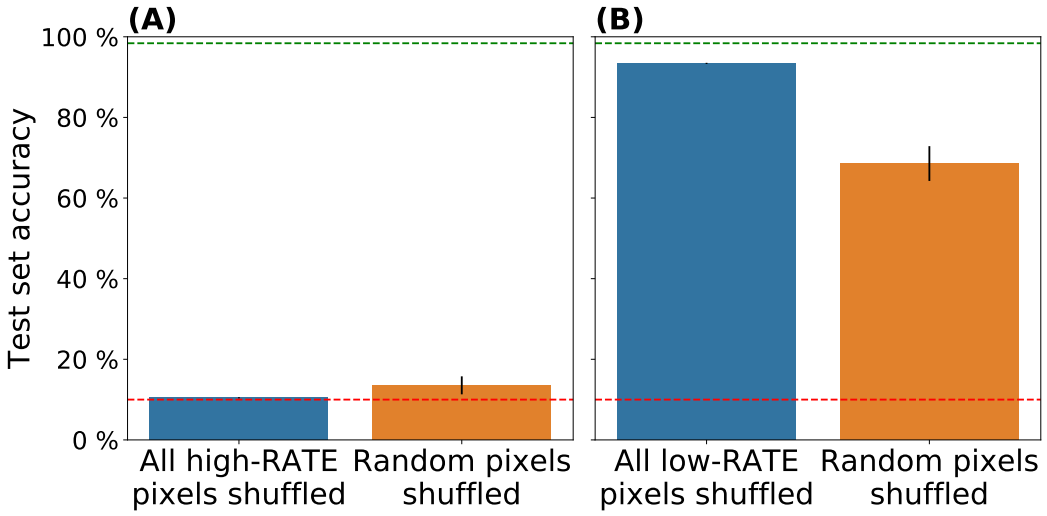

Figure 9: (A) Test set accuracies for MNIST when the 221 pixels with a RATE value greater than $1/p$ for any class are shuffled (blue), versus when the same number of any randomly-selected set of pixels are shuffled (orange). The images contain $p = 324$ pixels in total. Shuffling pixels with high RATE values reduces the test accuracy to that of a random classifier (red dotted line), while shuffling the same number of any randomly-selected set of pixels does not. (B) Test set accuracies for MNIST when the 103 pixels with a RATE value less than $1/p$ for every class are shuffled, versus when the same number of any randomly-selected set of pixels are shuffled. Compared to the accuracy on the unshuffled test set (green dotted line), shuffling the low-RATE pixels only reduces accuracy by 5.0%. Shuffling the randomly-selected pixels reduces the accuracy by 19.8%.

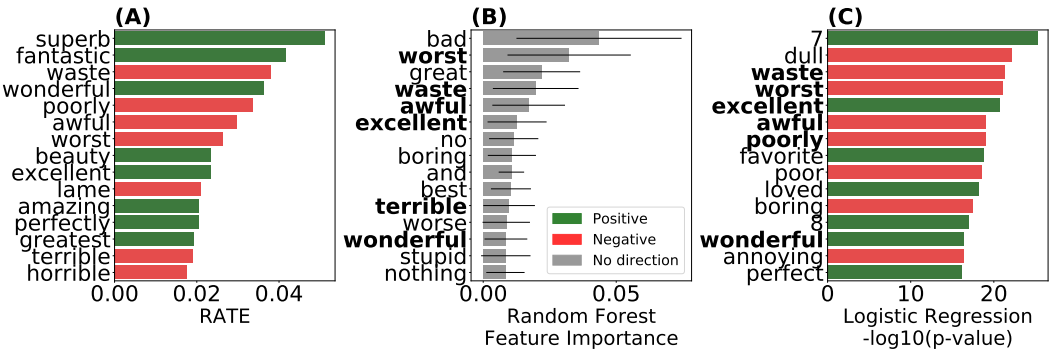

Figure 10: The 15 most important words for (A) a Bayesian neural network according to RATE, (B) a random forest classifier, and (C) logistic regression. The color of the bars indicates the direction of the variable's effect, which is taken from the sign of (A) the effect size analogue posterior mean, or (C) the sign of the regression coefficient. While this information is not available in (B), this plot does show $\pm$ the standard deviation of Gini importance across the trees of the random forest. Words in bold also appear in the top ranked list by the neural network and RATE.

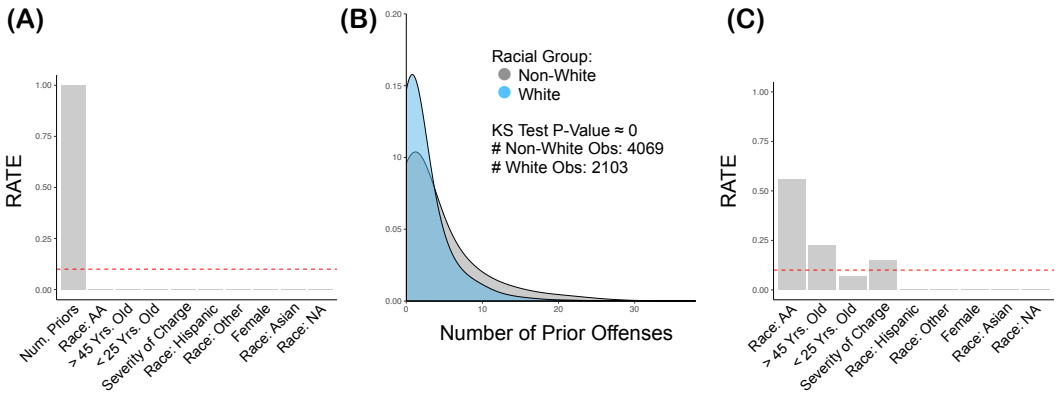

Figure 11: (A) First order RATE values from the Bayesian DNN. (B) Observed distribution of number of prior offenses by racial group. (C) First order RATE values from Bayesian DNN when number of prior offenses is omitted from the analysis. The dashed line is drawn at the level of relatively equal importance (i.e. $1/p$ and $1/(p-1)$ for panels (A) and (B), respectively).

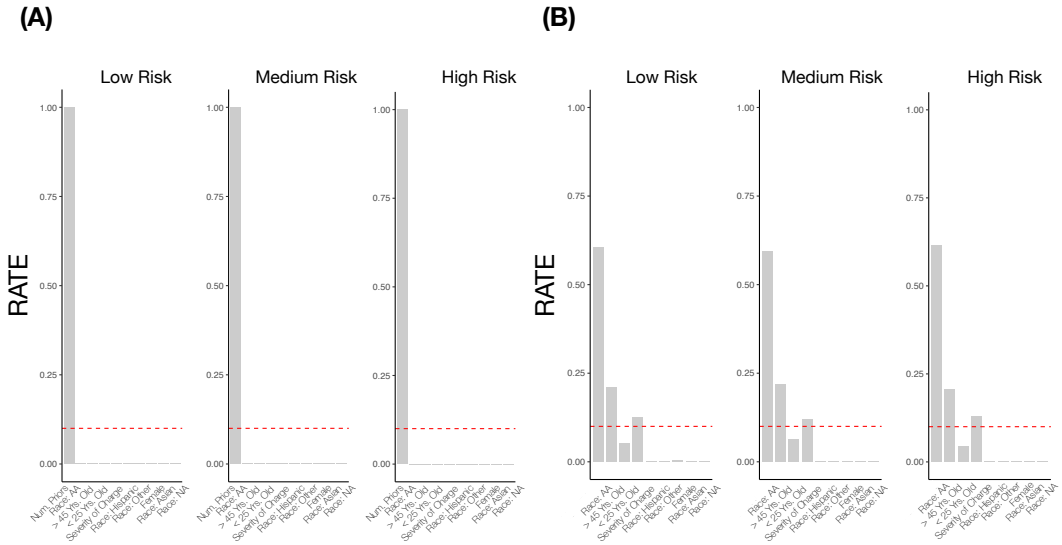

Figure 12: (A) First order RATE values from the Bayesian DNN. (B): First order RATE values from Bayesian DNN when number of prior offenses is omitted from the analysis. The dashed line is drawn at the level of relatively equal importance (i.e. $1/p$ and $1/(p-1)$ for panels (A) and (B), respectively).

Supplementary Tables

Table 1: Area under curves (AUCs) quantifying the ability of each variable importance method to identify truly causal variables in the simulation study. The values are the mean±standard deviation across 10 repeated experiments using datasets with n examples and p variables. For $n = 100,000$ the mimic models were not cross-validated (CV) as the running time was prohibitively long, so AUCs are not available.

n	VARIABLE IMPORTANCE METHOD	$p = 100$	$p = 300$	$p = 1,000$
1,000	RATE	0.98 ± 0.02	0.99 ± 0.01	0.99 ± 0.01
	RF MIMIC (NO CV)	0.90 ± 0.07	0.76 ± 0.03	0.57 ± 0.03
	RF MIMIC (WITH CV)	0.98 ± 0.03	0.93 ± 0.03	0.73 ± 0.05
	GBM MIMIC (NO CV)	0.95 ± 0.08	0.85 ± 0.04	0.66 ± 0.02
	GBM MIMIC (WITH CV)	0.96 ± 0.05	0.84 ± 0.05	0.66 ± 0.03
	CORRELATION	0.77 ± 0.10	0.75 ± 0.06	0.62 ± 0.02
10,000	RATE	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	RF MIMIC (NO CV)	1.00 ± 0.00	0.99 ± 0.01	0.82 ± 0.03
	RF MIMIC (WITH CV)	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	GBM MIMIC (NO CV)	1.00 ± 0.00	1.00 ± 0.00	0.93 ± 0.02
	GBM MIMIC (WITH CV)	1.00 ± 0.00	1.00 ± 0.00	0.98 ± 0.01
	CORRELATION	0.86 ± 0.08	0.86 ± 0.04	0.83 ± 0.03
100,000	RATE	0.99 ± 0.01	1.00 ± 0.01	1.00 ± 0.00
	RF MIMIC (NO CV)	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
	RF MIMIC (WITH CV)	N/A	N/A	N/A
	GBM MIMIC (NO CV)	1.00 ± 0.00	0.99 ± 0.01	0.91 ± 0.02
	GBM MIMIC (WITH CV)	N/A	N/A	N/A
	CORRELATION	0.75 ± 0.07	0.75 ± 0.06	0.75 ± 0.04

Table 2: Running time in seconds when computing variable importance for simulation studies with n examples and p variables. The cross-validation of the mimic models and RATE were calculated on 10 cores of a 2.1 GHz Intel Xeon Silver 4116 CPU. The other methods used a single core of this CPU. The running times are the mean $\pm$ standard deviation across ten repeated experiments. The pixel-wise correlation is not included as its running time is negligible. For $n = 100,000$ the mimic models were not cross-validated (CV) as the running time was prohibitively long.

n	VARIABLE IMPORTANCE METHOD	$p = 100$	$p = 300$	$p = 1,000$
1,000	RATE	0.5 ± 0.0	10.4 ± 0.1	325.2 ± 3.0
	RF MIMIC (NO CV)	0.1 ± 0.0	0.1 ± 0.0	0.1 ± 0.0
	RF MIMIC (WITH CV)	16.1 ± 2.3	23.2 ± 3.4	48.0 ± 10.2
	GBM MIMIC (NO CV)	0.6 ± 0.0	1.7 ± 0.1	5.5 ± 0.1
	GBM MIMIC (WITH CV)	13.4 ± 5.2	31.3 ± 14.0	74.6 ± 17.3
10,000	RATE	0.7 ± 0.0	10.6 ± 0.1	368.8 ± 1.1
	RF MIMIC (NO CV)	0.6 ± 0.0	1.0 ± 0.0	1.7 ± 0.0
	RF MIMIC (WITH CV)	298.2 ± 58.9	477.2 ± 91.3	864.7 ± 131.0
	GBM MIMIC (NO CV)	6.2 ± 0.2	18.7 ± 0.7	67.7 ± 0.6
	GBM MIMIC (WITH CV)	381.8 ± 93.1	796.2 ± 165.0	$2,335.8 \pm 930.8$
100,000	RATE	7.2 ± 0.1	19.0 ± 0.5	405.5 ± 12.1
	RF MIMIC (NO CV)	9.0 ± 0.3	16.4 ± 0.6	31.9 ± 0.3
	RF MIMIC (WITH CV)	N/A	N/A	N/A
	GBM MIMIC (NO CV)	91.4 ± 5.8	284.4 ± 7.5	$1,193.0 \pm 11.1$
	GBM MIMIC (WITH CV)	N/A	N/A	N/A

Table 3: Running time in seconds when computing variable importance on a binary classification task ($n = 12,665$ images and $p = 324$ pixels) using the digits 0 and 1 from MNIST. The experiments were repeated ten times and on a 2.1 GHz Intel Xeon Silver 4116 CPU. Cross-validation (CV) was performed in parallel using 10 cores, while the variable importance according to RATE and the mimics without cross-validation were calculated on a single core.

VARIABLE IMPORTANCE METHOD	TIME IN SECONDS (MEAN $\pm$ STANDARD DEVIATION)
RATE	5.8$\pm$ 0.1
RANDOM FOREST MIMIC (NO CV)	0.3$\pm$ 0.0
RANDOM FOREST MIMIC (WITH CV)	129.5$\pm$ 29.4
GRADIENT BOOSTING MACHINE MIMIC (NO CV)	18.0$\pm$0.4
GRADIENT BOOSTING MACHINE MIMIC (WITH CV)	270.8$\pm$ 69.2
LINEAR MIMIC (NO CV)	0.4 $\pm$ 0.0