INTERPRETABILITY WITH FULL COMPLEXITY BY CONSTRAINING FEATURE INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Interpretability is a pressing issue for machine learning. Common approaches to interpretable machine learning constrain interactions between features of the input, rendering comprehensible the effects of features on a model's output at the expense of model complexity. We approach interpretability from a different angle: constrain the information about the features utilized by the model, without any restrictions on the complexity of the model. Borrowing from information theory, we use the Distributed Information Bottleneck to find optimal compressions of each feature that maximally preserve information about the output. The learned information allocation, by feature and by feature value, is a rich source of interpretability well-suited to problems with many features and complex feature interactions. The central object of analysis is not a single trained model, but rather a spectrum of models serving as approximations that leverage variable amounts of information about the inputs. Information is allocated to features by their relevance to the output, tackling the problem of feature selection with inclusion/exclusion existing on a learned continuum. The optimal compression of each feature—at every stage of approximation—allows fine-grained inspection of how feature values are similar or distinct with regards to the prediction. We develop a framework for extracting insight from the spectrum of approximate models and demonstrate its utility on a range of tabular datasets.

1 INTRODUCTION

Interpretability is a pressing issue for machine learning (ML) (Doshi-Velez & Kim, 2017; Fan et al., 2021; Rudin et al., 2022). Models are growing more complex, and machine learning is increasingly integrated into fields where flawed decisions can have serious ramifications (Caruana et al., 2015; Rudin et al., 2018; Rudin, 2019).

Interpretability is not a binary property that machine learning methods have or do not: it is the degree to which a learning system can be probed and comprehended (Doshi-Velez & Kim, 2017), with many distinct routes to interpretability (Lipton, 2018). Various constraints on the learning system are incorporated that bring about a degree of comprehensibility (Molnar et al., 2020; Molnar, 2022). As opposed to explainable AI that happens post-hoc after a black-box model is trained, interpretable methods engineer the constraints into the model from the outset (Rudin et al., 2022). A common route to interpretability enforces that the effects of features combine in a simple (e.g. linear) manner, greatly restricting the space of possible models in exchange for comprehensibility.

In this work, we introduce a novel route to interpretability that places no restrictions on model complexity, and instead tracks *how much* and *what* information is most important for predictive power. By identifying optimal information from features of the input, the method grants a measure of salience of each feature, a spectrum of models utilizing different amounts of optimal information about the input, and a learned compression scheme for each feature that highlights in fine-grained detail from where the information is coming. The central object of interpretation is no longer a single optimized model, but rather a family of models revealing how predictive information is distributed across features at all levels of fidelity.

The information constraint we incorporate is a variant of the Information Bottleneck (IB) (Tishby et al., 2000; Asoodeh & Calmon, 2020). The IB is a framework for extracting the relevant information in a relationship, though it lacks interpretability because the extracted information is free be any

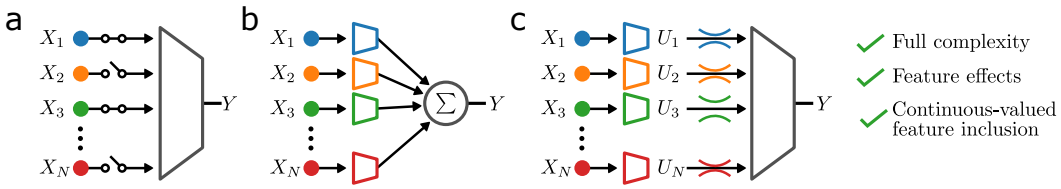

Figure 1: **Restricting information flow from features grants interpretability with full complexity. (a)** Feature selection methods optimize a binary mask on features to uncover the most informative ones. **(b)** Interpretable ML methods sacrifice model complexity in exchange for comprehensible feature interactions (shown is a GAM with a neural network to transform each feature). **(c)** Our method places an information bottleneck penalty on each feature, with no constraints on complexity before or after the bottlenecks. The features X_i are compressed to optimal representations U_i, which communicate the most relevant information to the rest of the model. Compressing the features in this way allows feature inclusion/exclusion to exist on a continuum. The learned compression scheme for each feature grants insight into feature effects on the model output.

black-box function of the input. Instead, by siloing the information from each feature and using a sum-rate constraint in the form of the Distributed IB (Estella Aguerri & Zaidi, 2018; Murphy & Bassett, 2022), we introduce a constraint at an appropriate level in the model's processing: after every feature is processed and before any interaction effects between the features can be included. Before and after the bottlenecks, arbitrarily complex processing can occur, allowing the Distributed IB to be used in scenarios with many features and complex feature interactions.

By restricting the flow of information into a model, we acquire a novel signal about the relationship in the data and the inner workings of the model. We develop a framework for extracting insight from the signal of interpretability acquired by the Distributed IB, and demonstrate the following strengths:

1. **Capacity to find the most important features *and* the most important feature values.** Interpretability comes from the fine-grained decomposition of the most relevant information in the data, found over the full spectrum of approximations and allowing a comprehensible entry point to analyzing the relationship.
2. **Full complexity of feature processing and interactions.** Compatible with arbitrary neural network architectures before and after the bottlenecks, the method gains interpretability without sacrificing predictive accuracy.
3. **Intuitive to implement, minimal additional overhead.** The method is a probabilistic encoder for each feature, employing the same loss penalty as β-VAE (Higgins et al., 2017). A single model is trained once while sweeping the magnitude of the information bottleneck.

We compare the Distributed IB to approaches in interpretable ML and feature selection on a variety of tabular datasets.

2 Related Work

We propose a method that analyzes the information content of features of an input with respect to an output, and its generality bears relevance to several broad areas of research.

Feature selection and variable importance. Feature selection methods isolate the most important features in a relationship (Fig. 1a), both for interpretability and to reduce computational burden (Chandrashekar & Sahin, 2014; Cai et al., 2018). When higher order feature interactions are present, the combinatorially large search space presents an NP-hard optimization problem (Chandrashekar & Sahin, 2014). Modern feature selection methods navigate the search space with feature inclusion weights, or masks, that can be trained as part of the full stack with the machine learning model (Fong & Vedaldi, 2017; Balın et al., 2019; Lemhadri et al., 2021; Kolek et al., 2022).

Many classic methods of statistical analysis work to identify the most important features in data. Canonical correlation analysis (CCA) (Hotelling, 1936) relates sets of random variables by finding transformations that maximize linear correlation. Modern extensions broaden the class of transformations to include kernel methods (Hardoon et al., 2004) and deep learning (Andrew et al., 2013; Wang

et al., 2016). Another line of research generalizes the linear correlation to use mutual information instead (Painsky et al., 2018). Analysis of variance (ANOVA) finds a decomposition of variance by features (Fisher, 1936; Stone, 1994); extensions incorporate ML-based transforms (Märtens & Yau, 2020). Variable importance (VI) methods quantify the effect of different features on an outcome (Breiman, 2001; Fisher et al., 2019).

Interpretable machine learning. Interpretability is often broadly categorized as *intrinsic*— the route to comprehensibility was engineered into the learning model from the outset, as in this work—or *post hoc*—constructed around an already-trained model (Rudin et al., 2022). Another common distinction is *local*—the comprehensibility built up piecemeal by inspecting individual examples (e.g., LIME (Ribeiro et al., 2016))—or *global*—the comprehensibility applies to the whole space of inputs at once, as in this work.

Intrinsic interpretabilty is generally achieved through constraints incorporated to reduce the complexity of the system. Generalized Additive Models (GAMs) require feature effects to combine linearly (Fig. 1b), though each effect can be an arbitrary function of a feature (Chang et al., 2021; Molnar, 2022). By using neural networks to learn the transformations from raw feature values to effects, Neural Additive Models (NAMs) maintain the interpretability of GAMs with more expressive effects (Agarwal et al., 2021). Pairwise interactions between effects can be included for more complex models (Yang et al., 2021; Chang et al., 2022), though complications arise involving degeneracy and the number of effect terms to inspect grows rapidly. Regularization can be used to encourage sparsity in the ML model so that fewer components—weights in the network (Ng, 2004) or even features (Lemhadri et al., 2021)—conspire to produce the final prediction.

Information Bottleneck. The proposed method can also be seen as a regularization to encourage sparsity of information. It is based upon the Information Bottleneck, a broad framework that extracts the most relevant information in a relationship (Tishby et al., 2000) and has proven useful in representation learning (Alemi et al., 2016; Wang et al., 2019; Asoodeh & Calmon, 2020). However, the standard IB has one bottleneck for the entire input, preventing an understanding of where the relevant information originated and hindering interpretability. Bang et al. (2021) and Schulz et al. (2020) employ the IB to explain a trained model's predictions as a form of post-hoc interpretability, whereas the current method is built into the training of a model and thus serves as intrinsic interpretability.

Distributing bottlenecks to multiple sources of data has been of interest in information theory as a solution to multiterminal source coding, or the "CEO problem" (Berger et al., 1996; Estella Aguerri & Zaidi, 2018; Aguerri & Zaidi, 2021; Steiner & Kuehn, 2021). Only recently has the Distributed IB been utilized for interpretability (Murphy & Bassett, 2022), whereby features of data serve as the distributed sources. With this formulation, the extracted information from each feature can be used to understand the role of the features in the relationship of interest.

3 Methods

At a high level, we introduce a penalty on the amount of information used by a black-box model about each feature of the input. Independently of all others, features are compressed and communicated through a noisy channel; the messages are then combined for input into a prediction network. The optimized information allocation and the learned compression schemes—for every feature and every magnitude of the penalty—serve as our windows into feature importance and into feature effects.

3.1 Distributed Variational Information Bottleneck

Let $X, Y \sim p(x, y)$ be random variables representing the input and output of a relationship to be modeled from data. The mutual information is a general measure of the statistical dependence of two random variables, and is the reduction of entropy of one variable after learning the value of the other:

$$I(X;Y) = H(Y) - H(Y|X), \tag{1}$$

with $H(Z) = \mathbb{E}_{z\sim p(z)}[-\log p(z)]$ Shannon's entropy (Shannon, 1948).

The Information Bottleneck (IB) (Tishby et al., 2000) is a learning objective to extract the most relevant information from X about Y. It optimizes a representation U that conveys information about X, realized as a stochastic channel of communication $U \sim p(u|x)$. The representation is found by minimizing a loss comprising competing mutual information terms balanced by a scalar parameter β,

$$\mathcal{L}_{\text{IB}} = \beta I(U;X) - I(U;Y). \tag{2}$$

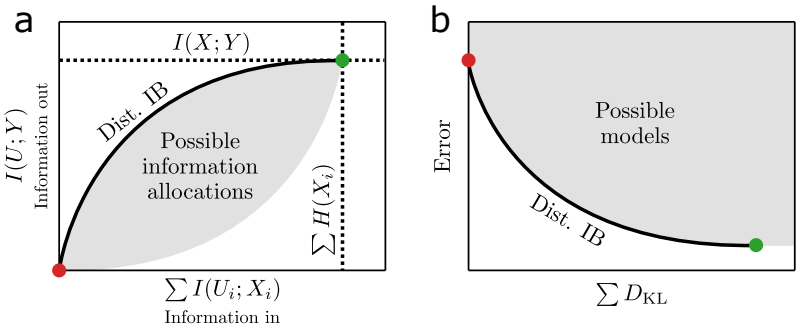

Figure 2: **The tradeoff between information into a model and predictive power out. (a)** The distributed information plane displays the sum total information from all features, encoded independently of one another, against the information about the output Y. While there are many possible allocations of feature information (gray region), the Distributed IB objective allows us to find a continuum of allocations (black curve) that contain the most information about the output for the least information about the input. Increasing the bottleneck strength β traverses the optimal trajectory from the green point—all bottlenecks fully open, and all information is available for prediction—to the red point—where predictions are independent of the input. **(b)** A pragmatic alternative to **(a)** displays the sum of the KL-divergences $\sum D_{\mathrm{KL}}$ (roughly corresponding to the information utilized by the model about the features) against the particular error metric being optimized. Optimizing the Distributed IB objective finds a continuum of models (black curve) that utilize the least information about the features for the least error. In the large $\sum D_{\mathrm{KL}}$ regime, the effect of the bottlenecks are negligible and the models are essentially unconstrained machine learning. A Distributed IB optimization gradually increases β, the strength of the bottlenecks, shedding information.

The first term is the bottleneck—acting as a penalty on information passing into U—with β determining the strength of the bottleneck. Optimization with different values of β produces different models of the relationship between X and Y, and a sweep over β produces a continuum of models that utilize different amounts of information about the input X. In the limit where $\beta \to 0$, the bottleneck is fully open and all information from X may be freely conveyed into U in order for it to be maximally informative about Y. As β increases, only the most relevant information in X about Y becomes worth conveying into U, until eventually it is optimal for U to be uninformative random noise.

Because mutual information is generally difficult to estimate from data (Saxe et al., 2019; McAllester & Stratos, 2020), variants of IB replace the mutual information terms of Eqn. 2 with bounds amenable to deep learning (Alemi et al., 2016; Achille & Soatto, 2018; Kolchinsky et al., 2019). We follow the Variational Information Bottleneck (VIB) (Alemi et al., 2016), which learns representations U in a framework resembling that of β-VAEs (Higgins et al., 2017).

A sample x is encoded as a distribution in representation space $p(u|x) = f(x, \phi, \epsilon)$ with a neural network parameterized by weights ϕ and a source of noise $\epsilon \sim \mathcal{N}(0, 1)$ to facilitate the "reparameterization trick" (Kingma & Welling, 2014). The bottleneck restricting the information from X takes the form of the Kullback-Leibler (KL) divergence—$D_{\mathrm{KL}}(p(z)||q(z)) = \mathbb{E}_{z\sim p(z)}[-\log\,(q(z)/p(z))]$—between the encoded distribution $p(u|x)$ and a prior distribution $r(u) = \mathcal{N}(0, 1)$. As the KL divergence tends to zero, all representations are indistinguishable from the prior and from each other, and therefore uninformative. A representation u is sampled from $p(u|x)$ and then decoded to a distribution over the output Y with a second neural network parameterized by weights ψ, $q(y|u) = g(u, \psi)$. In place of Eqn. 2, the following may be minimized with standard gradient descent methods:

$$\mathcal{L}_{\mathrm{VIB}} = \beta D_{\mathrm{KL}}(p(u|x)||r(u)) - \mathbb{E}[\log q(y|u)]. \tag{3}$$

The Distributed IB (Estella Aguerri & Zaidi, 2018) concerns the optimal scheme of compressing multiple sources of information as they relate to a single relevance variable Y. Let $\{X_i\}$ be the features of X; the Distributed IB encodes each feature X_i independently of the rest, and then integrates the information from all features to predict Y. A bottleneck is installed after each feature in the form of a compressed representation U_i (Fig. 1c), and the set of representations $U_X = \{U_i\}$ is

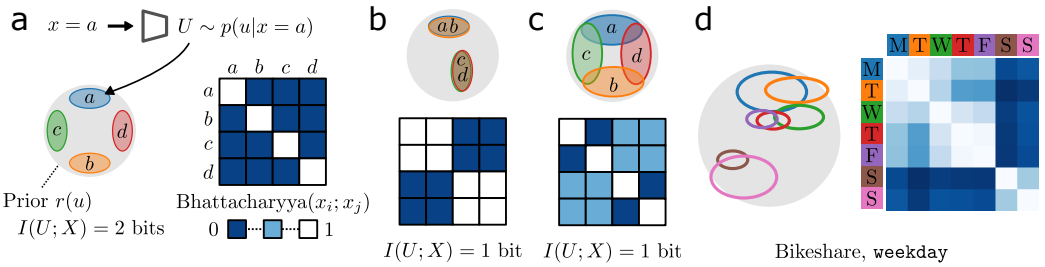

Figure 3: **Confusion matrices to visualize compression schemes**. **(a)** An input value is encoded to a distribution over u. The Bhattacharyya coefficient quantifies how distinguishable the distributions are for different encodings, which we express as a matrix. The diagonal (white) values are indistinguishable and everything else (blue) is distinguishable, yielding 2 bits of information. **(b)** 1 bit of information allows a hard clustering of the values, such that the ab cluster is distinguishable from the cd cluster, but no more than that. **(c)** 1 bit also allows a soft clustering, where some pairs of values are partially distinguishable. **(d)** When trained with a two-dimensional embedding space per feature on the Bikeshare dataset, we recover a soft clustering of the day of the week. Weekend is distinguishable from weekday, with intermediate values of distinguishability otherwise.

used as input to a model to predict Y. The same mutual information bounds of the Variational IB (Alemi et al., 2016) can be applied in the Distributed IB setting (Aguerri & Zaidi, 2021),

$$\mathcal{L}_{\text{DIB}} = \beta \sum_i D_{\text{KL}}(p(u_i|x_i)||r(u_i)) - \mathbb{E}[\log q(y|u_X)]. \tag{4}$$

Arbitrary error metrics with feature information regularization. Instead of quantifying predictive power with cross entropy, another error metric can be used in combination with the constraint on the information about the features. For instance, mean squared error (MSE) may be used instead,

$$\mathcal{L} = ||y - \hat{y}||^2 + \beta \sum D_{\text{KL}}, \tag{5}$$

with y and $\hat{y}$ the prediction and ground truth values, respectively. The connection to the IB is diminished, as β is no longer a unitless ratio of information about Y to the sum total information about the features of X, and the optimal range must be found anew for each problem. Optimization will nevertheless yield a spectrum of approximations prioritizing the most relevant information about the input features; the heart of the proposed framework is regularization on the information about input features, inspired by the Distributed Information Bottleneck rather than a strict application of it.

Boosting expressivity for low-dimensional features. Similarly with Agarwal et al. (2021), we found the feature encoders struggled with expressivity due to the low dimensionality of the features. We *positionally encode* feature values (Tancik et al., 2020) before they pass to the neural network, taking each value z to $[\sin(\omega_1 z), \sin(\omega_2 z), ...]$ where ω_k are the frequencies for the encoding.

3.2 Visualizing the optimally predictive information

Distributed information plane. Tishby & Zaslavsky (2015) popularized the visualization of learning dynamics in the "information plane", a means to display how successfully information conveyed about the input of a relationship is converted to information about the output. We employ the distributed information plane (Fig. 2a) to compare the total information utilized about the features $\sum I(U_i; X_i)$ to information about the output $I(\{U_i\}; Y)$. Considering the space of possible allocations of information to the input features, the Distributed IB finds the set of allocations that minimize $\sum I(U_i; X_i)$ while maximizing $I(\{U_i\}; Y)$. All possible allocations converge on two universal endpoints: the origin (no information in, no information out; red point in Fig. 2a) and at $(\sum H(X_i), I(X, Y))$ (all information in, mutual information between X and Y out; green point in Fig. 2a).

A pragmatic alternative to the distributed information plane replaces $\sum I(U_i; X_i)$ with its upper bound used during optimization, the sum of KL divergences across the features. Additionally, we replace the vertical axis $I(\{U_i\}; Y)$ with the error used for training. Switching the vertical axis from mutual information to an error has the effect of inverting the trajectory found by the Distributed IB: the uninformed point is now the *maximum* error with zero $\sum D_{\text{KL}}$ (red point in Fig. 2b).

Table 1: Performance on various tabular datasets. NAM and NODE-GAM allow no feature interactions; NODE-GA^{2}M contain pairwise feature interactions; the Neural Oblivious Decision Ensemble (NODE) LassoNet, and Distributed IB methods are full complexity models (allowing arbitrary feature interactions). ‡Results from Chang et al. (2022).

	Bikeshare (RMSE ↓)	Wine (RMSE ↓)	MIMIC-II (AUC % ↑)	MIMIC-III (AUC % ↑)	Microsoft (MSE ↓)
NAM (Agarwal et al., 2021)	99.9$\pm$1.3	0.71$\pm$0.02	83.0$\pm$0.8	81.7$\pm$0.2	0.5908$\pm$0.0004
NODE-GAM (Chang et al., 2022)	100.7$\pm$1.6	0.71$\pm$0.03	83.2$\pm$1.1	81.4$\pm$0.5	0.5821$\pm$0.0004
NODE-GA^{2}M (Chang et al., 2022)	49.8$\pm$0.8	0.67$\pm$0.02	84.6$\pm$1.1	82.2$\pm$0.7	0.5618$\pm$0.0003
LassoNet (Lemhadri et al., 2021)	46.0$\pm$2.0	0.68$\pm$0.02	83.4$\pm$0.8	79.4$\pm$0.8	0.5667$\pm$0.0012
NODE‡ (Popov et al., 2019)	**36.2**$\pm$1.9	0.64$\pm$0.01	84.3$\pm$1.1	**82.8**$\pm$0.7	**0.5570**$\pm$0.0002
Distributed IB (Ours)	38.0$\pm$1.1	**0.62**$\pm$0.01	**84.9**$\pm$1.2	80.2$\pm$0.4	0.5790$\pm$0.0003

The entire curve of optimal information allocations in Fig. 2b may be found in a single training run, by increasing β logarithmically over the course of training as Eqn. 4 is optimized. Training begins with negligible information restrictions ($\beta \sim 0$) so that the relationship may be found without obstruction (Wu et al., 2020; Lemhadri et al., 2021), and ends once all information is removed. Each value of β corresponds to a different approximation to the relationship between X and Y, and we can use the KL divergence for each feature as a measure of its overall importance in the approximation.

Confusion matrices to visualize feature compression schemes. Transmitting information about a feature is equivalent to conveying distinctions between the feature's possible values. In the course of optimizing the Distributed IB, the spectrum of models learned range from all input values perfectly distinguishable ($\beta \to 0$, unconstrained machine learning) to all inputs perfectly indistinguishable (the model utilizes zero information about the input). Along the way, the distinctions most important for predicting Y are selected, and can be inspected via the encoding of each feature value.

Each feature value x_i is encoded to a probability distribution $p(u_i|x_i)$ in channel space, allowing the compression scheme to be visualized with a confusion matrix showing the similarity of feature values (Fig. 3a). The Bhattacharyya coefficient (Kailath, 1967) is a symmetric measure of the similarity between probability distributions $p(z)$ and $q(z)$, and is straightforward to compute for normal distributions. The coefficient is 0 when the distributions are perfectly disjoint (i.e., zero probability is assigned by $p(z)$ everywhere that $q(z)$ has nonzero probability, and vice versa), and 1 when the distributions are identical. In other words, if the Bhattacharyya coefficient between encoded distributions $p(u|x = a)$ and $p(u|x = b)$ is 1, the feature values $x = a$ and $x = b$ are indistinguishable to the prediction network and the output is independent of whether the actual feature value was a or b. On the other extreme where the Bhattacharyya coefficient is 0, the feature values are perfectly distinguishable and the prediction network can yield distinct outputs for $x = a$ and $x = b$.

We display in Fig. 3b,c possible compression schemes containing one bit of information about a hypothetical x with four equally probable values. The compression scheme can be a hard clustering of the values into two sets where $x = a$ and $x = b$ are indistinguishable, and similarly for $x = c$ and $x = d$ (Fig. 3b). Another possibility with one bit of information is the soft clustering shown in Fig. 3c, where each feature value is partially 'confused' with other feature values. In the case of soft clustering, a learned compression cannot be represented by a decision tree; rather, the confusion matrices are the fundamental distillations of the information preserved about each feature. An example in Fig. 3d shows the learned compression scheme of the `weekday` feature in the Bikeshare dataset (dataset details below). The soft clustering distinguishes between the days of the week to different degrees, with the primary distinction between weekend and weekdays.

4 Experiments

We analyzed a variety of tabular datasets with the Distributed IB and compared the interpretability signal to what is obtainable through other methods. Our emphasis was on the unique capabilities of the Distributed IB to produce insight about the relationship in the data modelled by the neural network. Performance results are included in Tab. 1 for completeness. Note that the Distributed IB places no constraints on the machine learning models used and in the $\beta \to 0$ regime could hypothetically be as good as any finely tuned black-box model. The benefit of freedom from complexity constraints

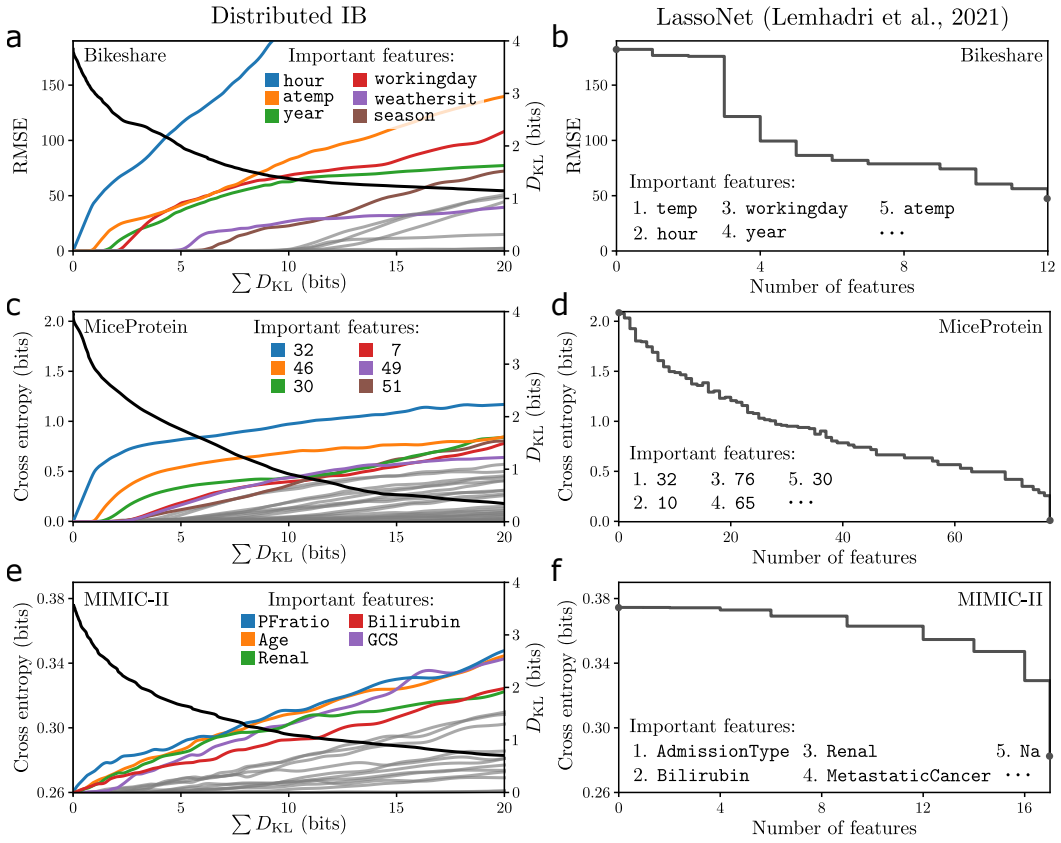

Figure 4: **Distributed information plane trajectories with information allocation by feature. (a)** For the Bikeshare dataset, the optimal RMSE error versus $\sum D_{\mathrm{KL}}$ (black) decays as more information is used by the model about the features. The shifting allocation of information to the features tells of their relevance to the prediction at each level of approximation. Only a few bits about a handful of features is enough for most of the performance. We label and color the first features to contribute non-negligibly to the approximations. **(b)** The LassoNet trajectory for the Bikeshare dataset yields only a discrete series of error values and an ordering of the features. Note the LassoNet trajectory extends to the point of all features included, whereas the Distributed IB is truncated at 20 bits of $\sum D_{\mathrm{KL}}$. **(c, d)** Same as **(a, b)** for the MiceProtein dataset, with expression levels of 77 proteins measured to classify mice in one of eight categories related to Down syndrome. **(e, f)** Same as **(a, b)** for the MIMIC-II dataset, a binary classification task related to ICU mortality, with 17 input features.

was particularly clear in the smaller regression datasets, whereas performance was only middling on several other datasets (refer to App. B for extended results and App. C for implementation details).

4.1 The features with the most predictive information

We used the Distributed IB to identify informative features for three datasets in Fig. 4. Bikeshare (Dua & Graff, 2017) is a regression dataset to predict the number of bikes rented given 12 descriptors of the date, time, and weather; MiceProtein (Higuera et al., 2015) is a classification dataset relating the expression of 77 genes to normal and trisomic mice exposed to various experimental conditions; MIMIC-II (Johnson et al., 2016) is a binary classification dataset to predict hospital ICU mortality given 17 patient measurements. We compared to LassoNet (Lemhadri et al., 2021), a feature selection method that attaches an L1 regularization penalty to each feature and introduces a proximal algorithm into the training loop to enforce a constraint on weights. By gradually increasing the regularization, LassoNet finds a series of models that utilize fewer features while maximizing predictive accuracy, providing a signal about what features are necessary for different levels of accuracy.

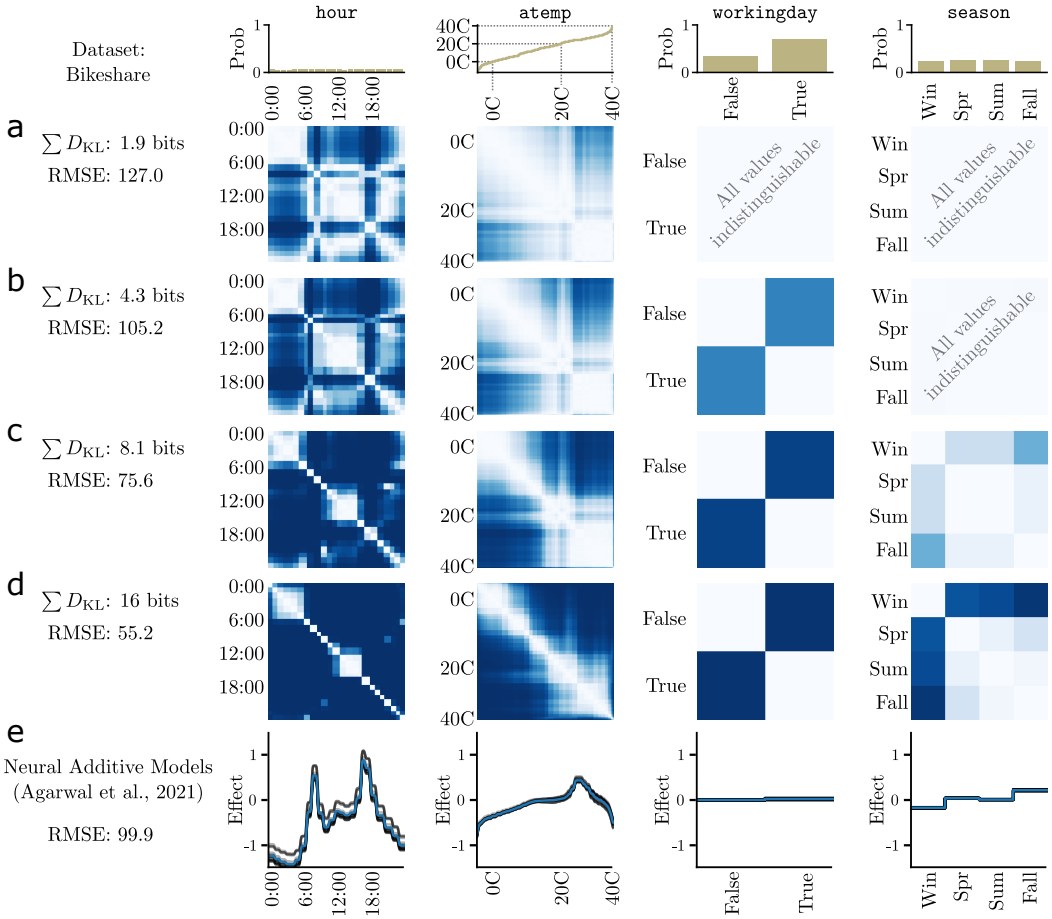

Figure 5: **Important information within each feature via inspection of the learned compression schemes. (a-d)** For the Bikeshare dataset at different levels of approximation, confusion matrices display the learned compression schemes for four of the top features found in Figure 4. Following the scheme in Fig. 3, the pairwise distinguishability of feature values visualizes the information preserved about each feature. For the categorical features (`hour`, `workingday`, and `season`), each row and column of the confusion matrices corresponds to a unique value, and the probability distributions at the top of the figure are aligned with the columns of the matrices. For the continuous variable `atemp` (apparent temperature), 1000 values were sampled randomly from the dataset and sorted for use as the inspected points; the rank order plot at the top shows their order in the matrices. As the utilized information $\sum D_{\mathrm{KL}}$ increases, so too does the performance; with only four bits the Distributed IB can perform about as well as NAM, highlighting the value of using a full-complexity model. **(e)** The effects of these features on the output can be directly visualized with NAM; the horizontal coordinates match those of the matrices in **(a-d)** and the black traces display effects for an ensemble of 25 NAMs.

Like LassoNet, the Distributed IB probes the tradeoff between information about the features and predictive performance, though with important differences. LassoNet can only vary the information used for prediction through a discrete-valued proxy: the number of features. Binary inclusion/exclusion of features is a common limitation for feature selection methods (Bang et al., 2021; Kolek et al., 2022). By contrast, the Distributed IB compresses each feature along a continuum, selecting the most informative information from across all input features. At every point along the continuum is an allocation of information to each of the features. For example, in Fig. 4a we found that only a few bits of information ($\sum D_{\mathrm{KL}}$), mostly coming from the `hour` feature, was necessary to achieve most of the performance of the full-information model, and to outperform the linear models (Tab. 1). The capacity to extract partial information from features shines in comparison to the LassoNet performance (Fig. 4b), which improves only after multiple whole features have been incorporated.

In contrast to Bikeshare, where decent approximations required only a handful of features, the MiceProtein (Fig. 4c,d) and MIMIC-II (Fig. 4e,f) datasets were found to require information from many features simultaneously. The information allocation is dynamic, with some features saturating quickly, and others growing in information usage at different rates. The signal obtained by the Distributed IB can help decompose datasets with a large number of features and complex interaction effects, all within a single training run.

4.2 Fine-grained inspection of feature effects

In addition to a measure of feature importance in the form of average information allocation, the Distributed IB offers a second source of interpretability: the relative importance of particular feature values, revealed through inspection of the learned compression schemes of each feature. One of the primary strengths of a GAM is that the effects of the feature values are straightforwardly obtained, though at the expense of interactions between the features. By contrast, the Distributed IB permits arbitrarily complex feature interactions, and still maintains a window into the role specific feature values play toward the ultimate prediction. In Fig. 5 we compared the compression schemes found by the Distributed IB to the feature effects obtained with NAM (Agarwal et al., 2021).

The Distributed IB allows one to build up to the maximally predictive model by approximations that gradually incorporate information from the input features. As an example, Fig. 5 displays approximations to the relationship between weather and bike rentals in the Bikeshare dataset. To a first approximation (Fig. 5a), with only two bits of information about all features, the scant distinguishing power was allocated to discerning certain times of day—commuting periods, the middle of the night, and everything else—in the `hour` feature. Some information discerned hot from cold from the apparent temperature (`atemp`), and no information about the season or whether it was a working day was used for prediction. We are unable to say how this information was used for prediction—the price of allowing the model to be more complex—but we can infer much from where the discerning power was allocated. At four bits, and performing about as well as the linear models in Tab. 1, the Boolean `workingday` was incorporated for prediction, and the evening commuting time was distinguished from the morning commuting time. With eight and sixteen bits of information about the inputs, nearly all of the information was used from the `hour` feature, the apparent temperature became better resolved, and in `season`, winter was distinguished from the warmer seasons.

The feature effects retrieved with NAM (Fig. 5e) corroborated some of the distinguishability found by the Distributed IB, particularly in the `hour` feature (e.g., the commuting times, the middle of the night, and everything else form three characteristic divisions of the day in terms of effect). However, we found the interpretability of the NAM-effects to vary significantly across datasets (see App. B for more examples). For example, the apparent temperature (`atemp`) and temperature (`temp`) features were found to have contradictory effects (Fig. 10) that were difficult to comprehend, presumably arising from feature interactions that NAM could not model. Thus while the Distributed IB sacrifices a direct window to feature effects, tracking the information on the input side of a black-box model may serve as one of the only sources of interpretability in cases with complex feature interactions.

5 Discussion

Interpretability can come in many forms (Lipton, 2018); the Distributed IB provides a novel sources of interpretability by illuminating where information resides in a relationship with regards to features of the input. Without having to sacrifice complexity, the Distributed IB is particularly useful in scenarios where obtaining a foothold is difficult due to large numbers of features and complex interaction effects. The capacity for successive approximation, allowing detail to be incorporated gradually so that humans' limited cognitive capacity (Cowan, 2010; Rudin et al., 2022) does not prevent comprehension of the full relationship, plays a central role in the Distributed IB's interpretability.

The signal from the Distributed IB—particularly the compression schemes of feature values (e.g. Fig. 5)—is vulnerable to over-interpretation. This is partly due to a difficulty surrounding intuition about compression schemes (i.e., it is perhaps most natural to imagine 1 bit of information as a single yes/no question, or a hard clustering (Fig. 3b), rather than as a confusion matrix (Fig. 3c)), and partly due to the inability to make claims like "changing input A to input B changes output C to output D". Rather than supplanting other interpretability methods, the Distributed IB should be seen as a new tool for interpretable machine learning, supplementing other methods. A thorough analysis could incorporate methods where the effects are comprehensible, possibly using a reduced feature set found by the Distributed IB.

REFERENCES

Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40 (12):2897–2905, 2018. 4

Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. Neural additive models: Interpretable machine learning with neural nets. *Advances in Neural Information Processing Systems*, 34:4699–4711, 2021. 3, 5, 6, 9, 20

Iñaki Estella Aguerri and Abdellatif Zaidi. Distributed variational representation learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):120–138, 2021. doi: 10.1109/TPAMI.2019.2928806. 3, 5

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016. 3, 4, 5

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pp. 1247–1255. PMLR, 2013. 2

Shahab Asoodeh and Flavio P. Calmon. Bottleneck problems: An information and estimation-theoretic view. *Entropy*, 22(11):1325, 2020. 1, 3

Muhammed Fatih Balın, Abubakar Abid, and James Zou. Concrete autoencoders: Differentiable feature selection and reconstruction. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 444–453. PMLR, 09–15 Jun 2019. 2

Seojin Bang, Pengtao Xie, Heewook Lee, Wei Wu, and Eric Xing. Explaining a black-box by using a deep variational information bottleneck approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11396–11404, 2021. 3, 8

Toby Berger, Zhen Zhang, and Harish Viswanathan. The CEO problem [multiterminal source coding]. *IEEE Transactions on Information Theory*, 42(3):887–902, 1996. 3

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 3

Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79, 2018. 2

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noémie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1721–1730, 2015. 1

Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014. 2

Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How interpretable and trustworthy are gams? In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 95–105, 2021. 3

Chun-Hao Chang, Rich Caruana, and Anna Goldenberg. Node-gam: Neural generalized additive model for interpretable deep learning. In *International Conference on Learning Representations*, 2022. 3, 6, 15, 18

Nelson Cowan. The magical mystery four: How is working memory capacity limited, and why? *Current Directions in Psychological Science*, 19(1):51–57, 2010. 9

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. 1

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml. 7, 15

Iñaki Estella Aguerri and Abdellatif Zaidi. Distributed information bottleneck method for discrete and gaussian sources. In *International Zurich Seminar on Information and Communication (IZS 2018). Proceedings*, pp. 35–39. ETH Zurich, 2018. 2, 3, 4

Feng-Lei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, 5(6):741–760, 2021. 1

Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *J. Mach. Learn. Res.*, 20(177):1–81, 2019. 3

Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7 (2):179–188, 1936. 3

Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pp. 3429–3437, 2017. 2

David R. Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639–2664, 2004. 2

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations (ICLR)*, 2017. 2, 4

Clara Higuera, Katheleen J Gardiner, and Krzysztof J Cios. Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome. *PloS one*, 10(6):e0129126, 2015. 7

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 12 1936. ISSN 0006-3444. doi: 10.1093/biomet/28.3-4.321. URL https://doi.org/10.1093/biomet/28.3-4.321. 2

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016. 7, 15

Thomas Kailath. The divergence and bhattacharyya distance measures in signal selection. *IEEE transactions on communication technology*, 15(1):52–60, 1967. 6

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 4

Artemy Kolchinsky, Brendan D. Tracey, and David H. Wolpert. Nonlinear information bottleneck. *Entropy*, 21(12), 2019. ISSN 1099-4300. doi: 10.3390/e21121181. URL https://www.mdpi.com/1099-4300/21/12/1181. 4

Stefan Kolek, Duc Anh Nguyen, Ron Levie, Joan Bruna, and Gitta Kutyniok. A rate-distortion framework for explaining black-box model decisions. In *International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers*, pp. 91–115. Springer, 2022. 2, 8

Ismael Lemhadri, Feng Ruan, Louis Abraham, and Robert Tibshirani. Lassonet: A neural network with feature sparsity. *Journal of Machine Learning Research*, 22(127):1–29, 2021. URL http://jmlr.org/papers/v22/20-848.html. 2, 3, 6, 7, 20

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018. 1, 9

Kaspar Märtens and Christopher Yau. Neural decomposition: Functional ANOVA with variational autoencoders. In *International Conference on Artificial Intelligence and Statistics*, pp. 2917–2927. PMLR, 2020. 3

David McAllester and Karl Stratos. Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*, pp. 875–884. PMLR, 2020. 4

Christoph Molnar. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2022. 1, 3

Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. Interpretable machine learning–a brief history, state-of-the-art and challenges. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 417–431. Springer, 2020. 1

Kieran A Murphy and Dani S Bassett. The distributed information bottleneck reveals the explanatory structure of complex systems. *arXiv preprint arXiv:2204.07576*, 2022. 2, 3

Andrew Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 78, 2004. 3

Amichai Painsky, Meir Feder, and Naftali Tishby. An information-theoretic framework for non-linear canonical correlation analysis. *CoRR*, 2018. 3

Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312*, 2019. 6

Tao Qin and Tie-Yan Liu. Introducing LETOR 4.0 datasets. *CoRR*, abs/1306.2597, 2013. URL http://arxiv.org/abs/1306.2597. 15

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016. 3

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. 1

Cynthia Rudin, Caroline Wang, and Beau Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv preprint arXiv:1811.00731*, 2018. 1

Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–85, 2022. 1, 3, 9

Andrew M. Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D. Tracey, and David D. Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, dec 2019. doi: 10.1088/1742-5468/ab3985. URL https://doi.org/10.1088%2F1742-5468%2Fab3985. 4

Karl Schulz, Leon Sixt, Federico Tombari, and Tim Landgraf. Restricting the flow: Information bottlenecks for attribution. *arXiv preprint arXiv:2001.00396*, 2020. 3

Claude Elwood Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. 3

Steffen Steiner and Volker Kuehn. Distributed compression using the information bottleneck principle. In *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021. doi: 10.1109/ICC42927.2021.9500324. 3

Charles J. Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *The Annals of Statistics*, 22(1):118–171, 1994. 3

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 5

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2015. 5

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000. 1, 3

Qi Wang, Claire Boudreau, Qixing Luo, Pang-Ning Tan, and Jiayu Zhou. Deep multi-view information bottleneck. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 37–45. SIAM, 2019. 3

Weiran Wang, Xinchen Yan, Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*, 2016. 2

Tailin Wu, Ian Fischer, Isaac L. Chuang, and Max Tegmark. Learnability for the information bottleneck. In *Uncertainty in Artificial Intelligence*, pp. 1050–1060. PMLR, 2020. 6

Zebin Yang, Aijun Zhang, and Agus Sudjianto. Gami-net: An explainable neural network based on generalized additive models with structured interactions. *Pattern Recognition*, 120:108192, 2021. 3

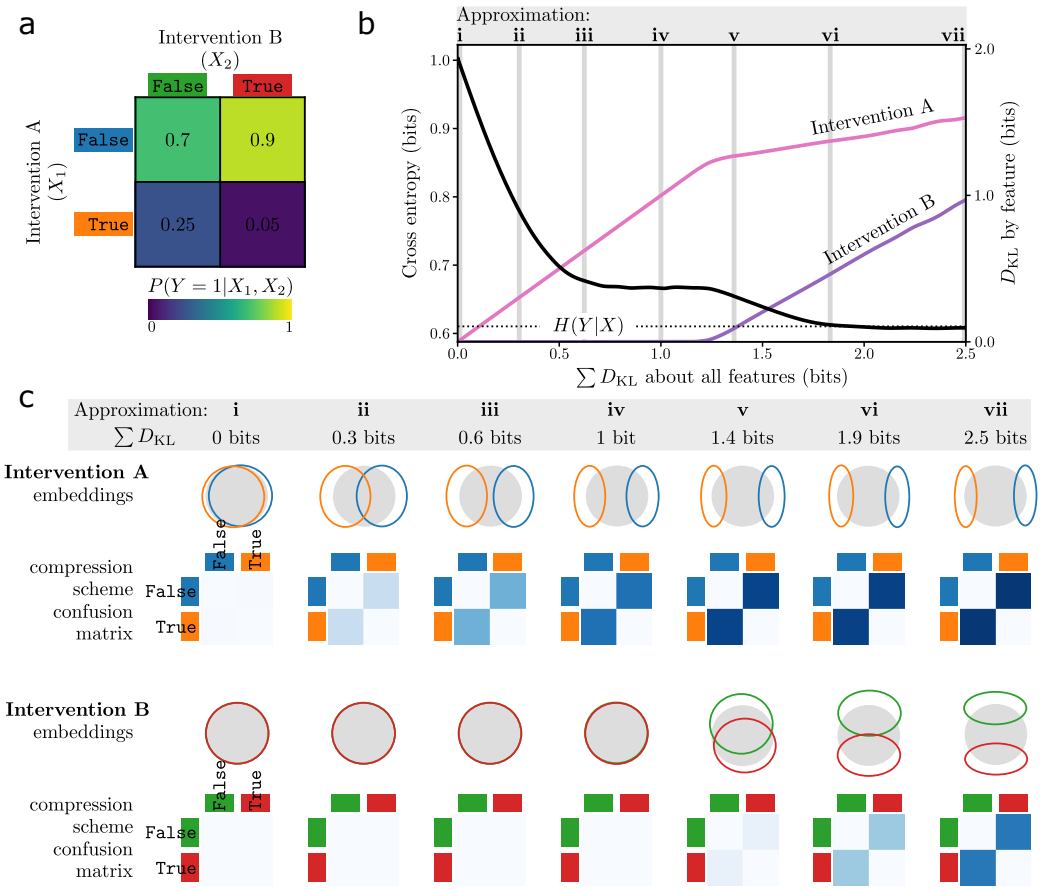

Figure 6: **Synthetic example: two interventions on a binary-valued outcome.** We visualize information allocation on a simple example. **(a)** Consider a hypothetical situation with two possible interventions, A and B, applied uniformly at random and affecting a binary outcome Y. The effects of applying the interventions can be visualized by a 2×2 table displaying $P(Y=1|X)$. **(b)** By training the Distributed IB with 10,000 samples from the joint distribution represented by the table in **(a)**, we obtain a trajectory in the distributed information plane (black) showing the cross entropy error as a function of the amount of information incorporated about the interventions. Intervention A is found to be more important in predicting the outcome of Y. When information about intervention B is incorporated, the error eventually bottoms out at the conditional entropy $H(Y|X)$. **(c)** With each feature value embedded to a two-dimensional Gaussian distribution, we can visualize the embeddings alongside the computed confusion matrices. For the seven approximations highlighted in **(b)**, we display the learned compression schemes for interventions A (top) and B (bottom). As noted in the information allocation in **(b)**, the feature values of intervention A are distinguished first, and then the values of intervention B around approximation **v** and beyond.

A Synthetic Example

We work through a synthetic example with the Distributed IB in Fig. 6. There are two binary-valued interventions A, B which we take to be the features X_1, X_2 of the input X, affecting a binary outcome Y with a joint probability distribution shown in Fig. 6a. Using 10,000 samples from the joint distribution, we train the Distributed IB and recover the distributed information plane (Fig. 6b) revealing that intervention A is more important for predicting the outcome Y.

Each feature is embedded to a two-dimensional Gaussian so that they may be directly visualized (Fig. 6c), alongside the computed confusion matrix. Note that the feature values of the interventions

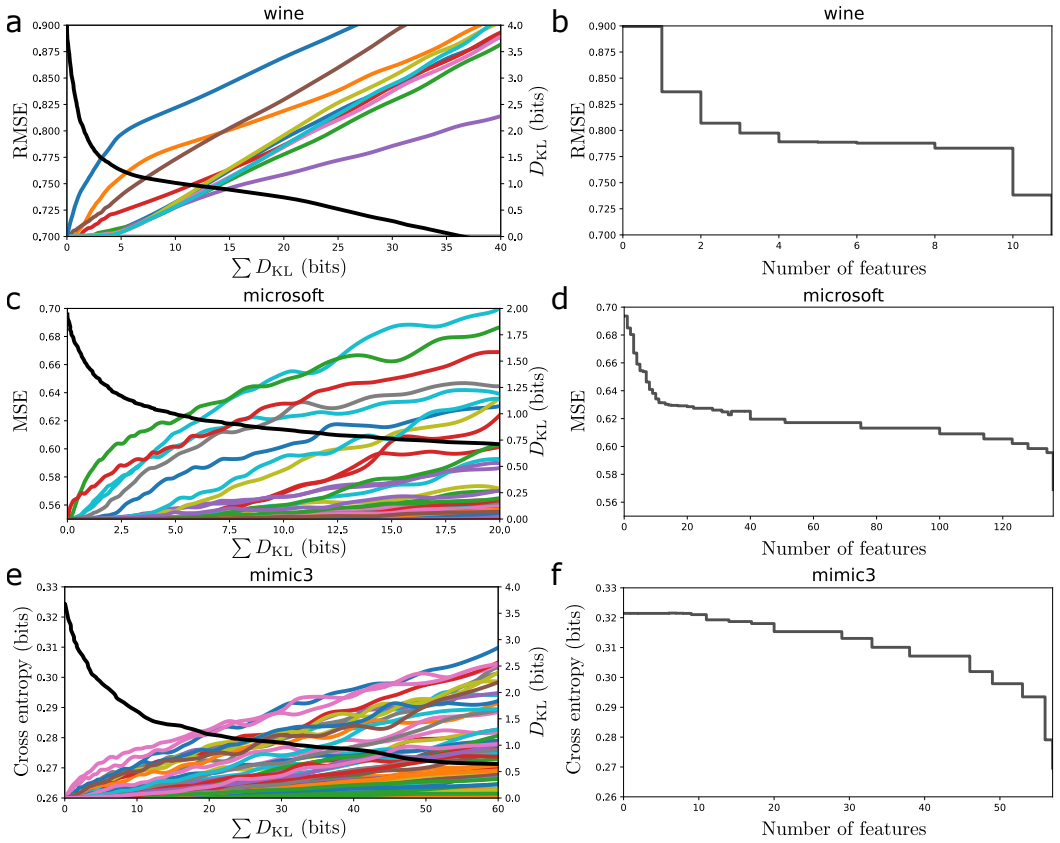

Figure 7: **Information plane optimizations for the remaining datasets.** We display Distributed IB optimizations alongside LassoNet results. Note the $\sum D_{\mathrm{KL}}$ range is not uniform across datasets. **(a, b)** Wine, regress wine quality with 12 features (Dua & Graff, 2017); **(c, d)** Microsoft Learn to Rank, 136 features (Qin & Liu, 2013); **(e, f)** MIMIC-III, similar to MIMIC-II but with 57 features (Johnson et al., 2016).

split along the cardinal axes because the embeddings are diagonal Gaussians. That intervention A split horizontally while B split vertically is due to chance.

B Extended visualizations

We show in Fig. 7 Distributed IB optimizations for the remaining three datasets mentioned in the manuscript, alongside their LassoNet counterparts. In Figs. 8 and 9 we show the compression schemes for some of the most important features in the Wine and MIMIC-II datasets. Finally, in Fig. 10, we display the compression schemes for all 12 features of the Bikeshare dataset over a dense sampling of information rates.

C Implementation details

All Distributed IB experiments were run on a single computer with a 12 GB GeForce RTX 3060 GPU; 100k steps on the Bikeshare dataset takes a couple minutes, and on Microsoft about 2 hours.

Tensorflow code is attached to this submission and will be posted (in extended form) to Github upon acceptance. The data loading and set up is heavily based on the publically released code from NODE-GAM (Chang et al., 2022).

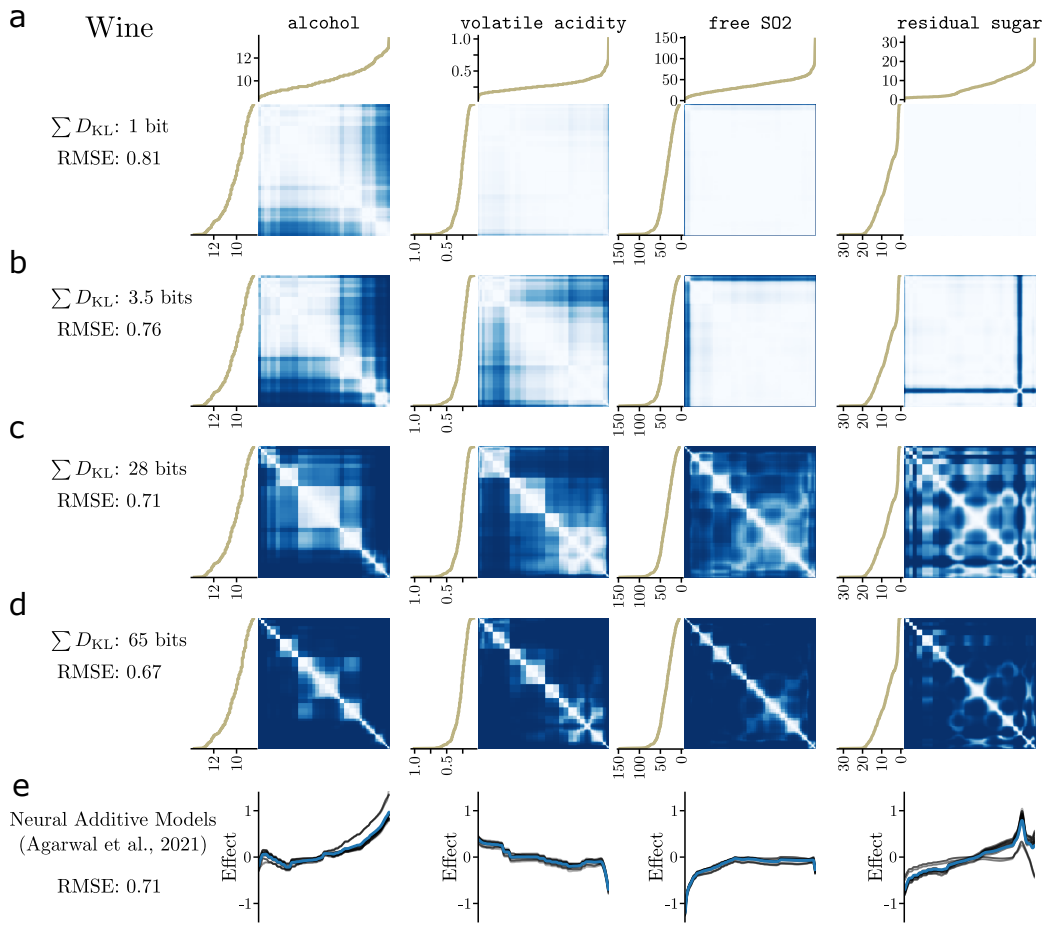

Figure 8: **Feature effects: Wine. (a-d)** For the Wine dataset, we display distinguishability matrices at different levels of approximation for four features. To the left (top) of the distinguishability matrices are the feature values corresponding to the row (column), which have been sorted. The features (`alcohol`, `volatile acidity`, `free SO2`, and `residual sugar`) were the first four to contribute to the approximations in the low $\sum D_{\mathrm{KL}}$ regime of Fig. 7a. As $\sum D_{\mathrm{KL}}$ increases, the RMSE improves. **(e)** The effects of these four features can be directly visualized with NAM; the horizontal coordinates match those of the matrices in **(a-d)** and the black traces display effects for an ensemble of 25 NAMs.

We subclassed `tensorflow.keras.Model` so that training can leverage the `keras` high level API, and is only one step more difficult than the `model.compile(); model.fit()` flow. The additional step is the creation of a `tensorflow.keras.callbacks.Callback` to control the β ramp over the course of training. All of the information allocations can be recovered from the `tensorflow.keras.callbacks.History` object returned at the end of training.

Training hyperparameters and architecture details are shown in Tab. 2 (common to all datasets) and Tab. 3 (dataset-specific). In the first stage of hyperparameter tuning, we explored parameter space in an unstructured manner and found that most parameters affected all datasets similarly. After some tuning, we settled on the values listed in Tab. 2. We found the effects of dropout rate and the number of annealing steps to differ by dataset, and selected optimal values out of $\{0, 0.1, 0.3, 0.5\}$ and $\{2.5 \times 10^4, 5 \times 10^4, 10^5\}$, respectively (Tab. 3).

Parameter	Value
Positional encoding frequencies	[1, 2, 4, 8]
Nonlinear activation	Leaky ReLU ($\alpha = 0.2$)
Feature encoder MLP (one per feature)	[128, 128]
Bottleneck embedding space dimension	8
Joint encoder MLP	[256, 256]
Batch size	128
Optimizer	Adam
Learning rate	3×10^{-4}
β_{initial}	2×10^{-5}
β_{final}	2

Table 2: Training parameters for the Distributed IB experiments.

Parameter	Dataset	Value
Annealing steps	Bikeshare	10^5
	Wine	10^5
	MIMIC 2	5×10^4
	MIMIC 3	5×10^4
	Mice protein	5×10^4
	Microsoft	10^5
Dropout fraction	Bikeshare	0
	Wine	0.1
	MIMIC 2	0.3
	MIMIC 3	0.5
	Mice protein	0.1
	Microsoft	0.1

Table 3: Dataset-specific parameters for the Distributed IB experiments.

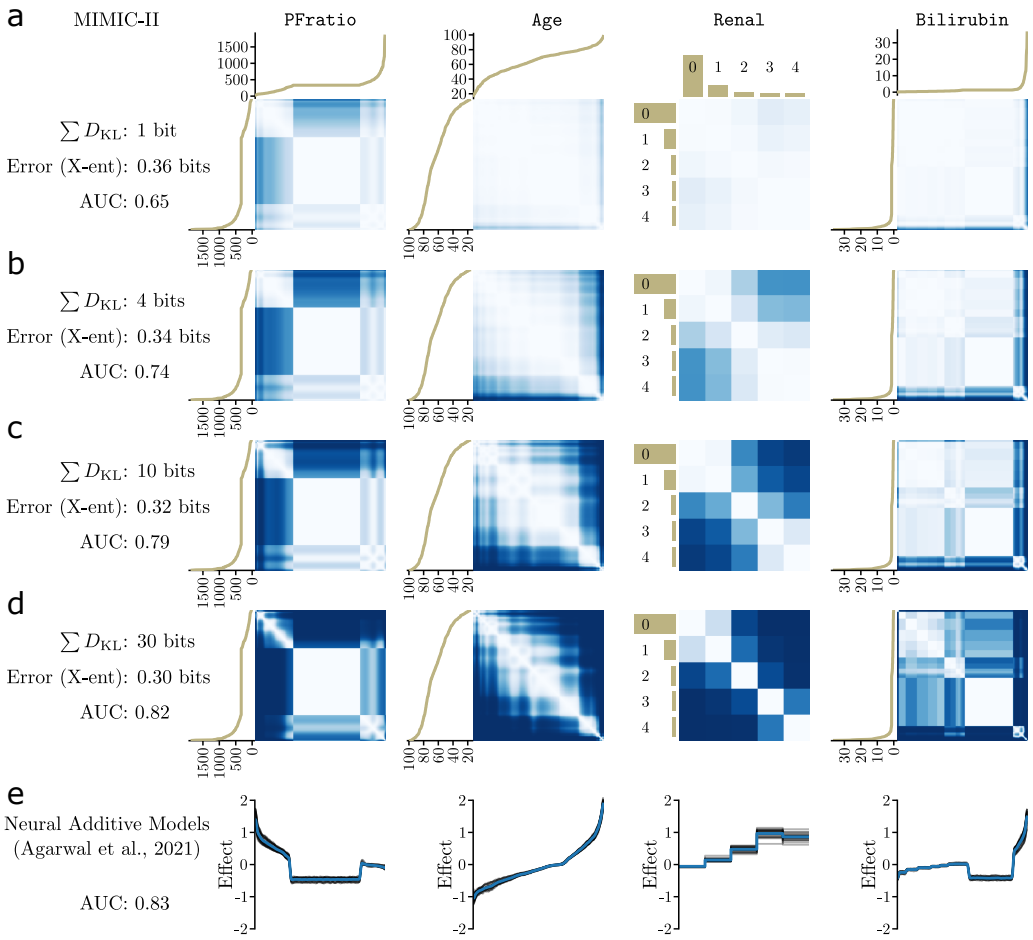

Figure 9: **Feature effects: MIMIC-II. (a-d)** For the MIMIC-II dataset, we display distinguishability matrices at different levels of approximation for four features. To the left (top) of the distinguishability matrices are the feature values corresponding to the row (column), which have been sorted. The features (`PFratio`, `Age`, `Renal`, and `Bilirubin`) were the first four to contribute to the approximations in the low $\sum D_{\mathrm{KL}}$ regime of Fig. 4. For the categorical variable `Renal` the barchart displays the frequency of the particular values. As $\sum D_{\mathrm{KL}}$ increases, the cross entropy error decreases and the ROC AUC (area under curve) increases. **(e)** The effects of these four features can be directly visualized with NAM; the horizontal coordinates match those of the matrices in **(a-d)** and the black traces display effects for an ensemble of 25 NAMs.

C.1 Datasets

We used the dataset preprocessing code released with NODE-GAM (Chang et al., 2022) on Github[1] with a minor modification to use one-hot encoding[2] for categorical variables instead of leave-one-out encoding[3] when input to the Distributed IB models, unless there were more than 100 categories.

[1] https://github.com/zzzace2000/nodegam
[2] https://contrib.scikit-learn.org/category_encoders/onehot.html
[3] https://contrib.scikit-learn.org/category_encoders/leaveoneout.html

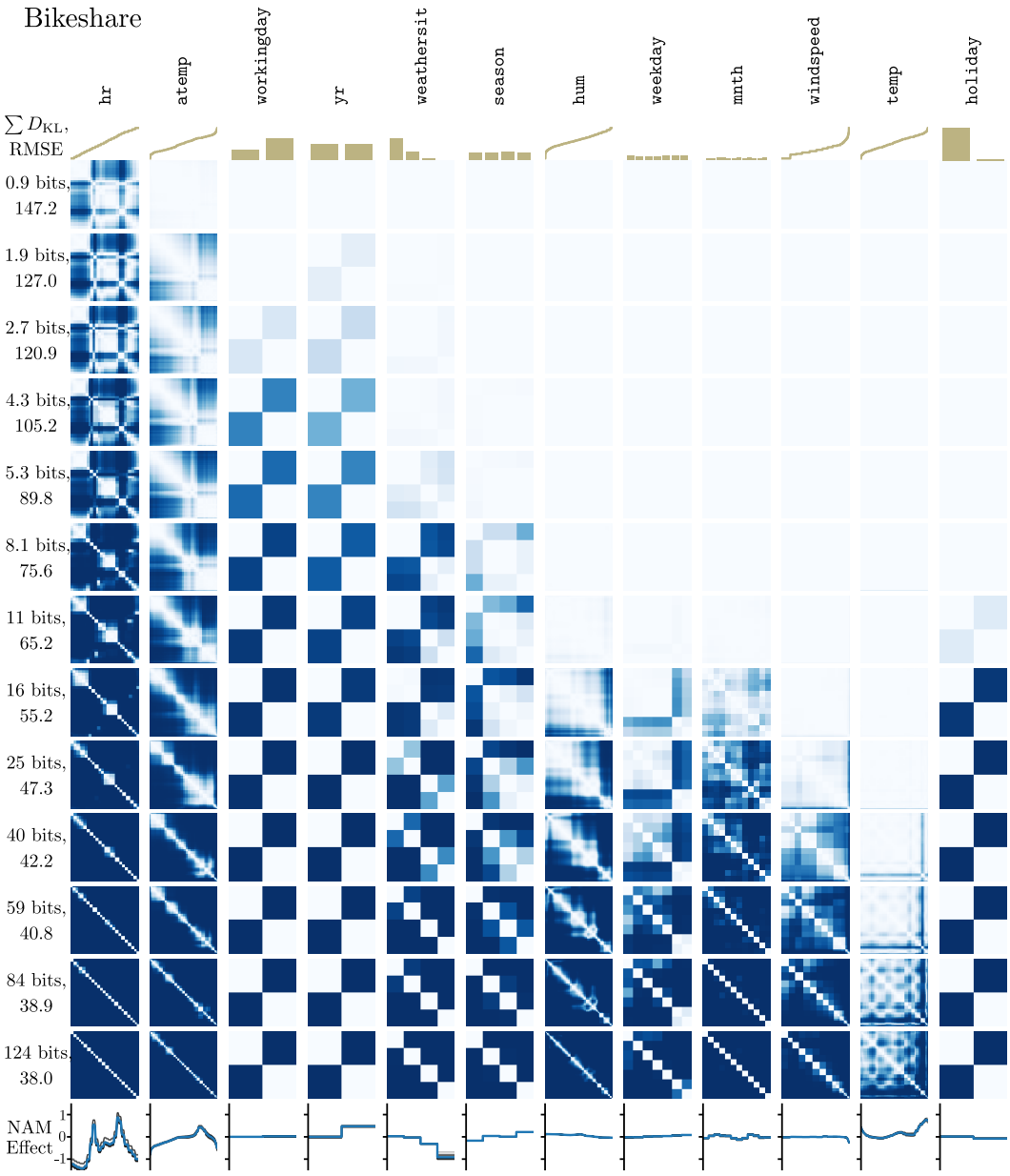

Figure 10: **Feature compression schemes: Bikeshare, extended.** We show the compression schemes for all 12 features of the Bikeshare dataset, for many values of total D_{KL}, and compare in the bottom row to the feature effects found by NAM. The features were ordered left to right by the order of crossing a threshold value in Figure 4. Note the gradual addition of features and of distinguishability as the information rate increases. Eventually most of the feature value confusion matrices tend toward diagonal matrices, signifying that each unique feature value is distinguishable from all others in the compression scheme.

C.2 COMPARISON METHODS

LassoNet (Lemhadri et al., 2021) We used the publicly released Pytorch code[4] with only minor modifications: (i) fixed a typo in `interfaces.py`[5] (delete '='), (ii) changed the objective outputs during training[6] to be solely the error metric (excluding the regularization terms) for the plots of Fig. 4.

We used the `LassoNetClassifier` and `LassoNetRegressor` classes, instead of the cross-validation versions that were released after the original publication, and performed our own cross-validation with the same splits for the other methods. For each dataset we tuned the following hyperparameters with a grid search: L2-regularization weight $\gamma \in \{0, 0.01, 0.1, 1\}$ and the dropout fraction in $\{0, 0.1, 0.3, 0.5\}$. We selected the parameters with the best integrated performance on the validation set (metric $\times$ number of features), and used the same hidden dimensions as the joint encoder for the Distributed IB method (2 layers of 256 units).

Neural additive models (Agarwal et al., 2021) We used the publicly released Tensorflow code[7]. We modified the size of the feature encoders: we encountered memory issues with the default feature encoder sizes of [M, 64, 32], with M a function of the number of unique values in the training set for that feature[8]. Instead we made each feature encoder size [256, 256, 256] for the datasets with under 20 features, and [128, 128, 128] for the rest.

For each dataset we tuned the following hyperparameters with a grid search: hidden activation in $\{\texttt{relu}, \texttt{exu}\}$, dropout in $\{0, 0.1, 0.3\}$, weight decay (L2 regularization) in $\{0, 10^{-5}, 10^{-4}\}$.

[4] https://github.com/lasso-net/lassonet

[5] https://github.com/lasso-net/lassonet/blob/54d05048eca51c7b57077f0e09d4a72767f3e567/lassonet/interfaces.py#L470

[6] https://github.com/lasso-net/lassonet/blob/54d05048eca51c7b57077f0e09d4a72767f3e567/lassonet/interfaces.py#L251

[7] https://github.com/google-research/google-research/tree/master/neural_additive_models

[8] https://github.com/google-research/google-research/blob/e79b8f09344935a2ab22660293233eb032df13a1/neural_additive_models/graph_builder.py#L283