entropy

Article

Nonlinear Information Bottleneck

Artemy Kolchinsky [1,*], Brendan D. Tracey [1,2] and David H. Wolpert [1,3,4]

1 Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501, USA; tracey.brendan@gmail.com (B.D.T.); david.h.wolpert@gmail.com (D.H.W.)
2 Department of Aeronautics & Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
3 Complexity Science Hub, 1080 Vienna, Austria
4 Center for Bio-Social Complex Systems, Arizona State University, Tempe, AZ 85281, USA
* Correspondence: artemyk@gmail.com

Received: 16 October 2019; Accepted: 28 November 2019; Published: 30 November 2019

Abstract: Information bottleneck (IB) is a technique for extracting information in one random variable X that is relevant for predicting another random variable Y. IB works by encoding X in a compressed "bottleneck" random variable M from which Y can be accurately decoded. However, finding the optimal bottleneck variable involves a difficult optimization problem, which until recently has been considered for only two limited cases: discrete X and Y with small state spaces, and continuous X and Y with a Gaussian joint distribution (in which case optimal encoding and decoding maps are linear). We propose a method for performing IB on arbitrarily-distributed discrete and/or continuous X and Y, while allowing for nonlinear encoding and decoding maps. Our approach relies on a novel non-parametric upper bound for mutual information. We describe how to implement our method using neural networks. We then show that it achieves better performance than the recently-proposed "variational IB" method on several real-world datasets.

Keywords: information bottleneck; mutual information; representation learning; neural networks

1. Introduction

Imagine that one has two random variables, an "input" random variable X and an "output" random variable Y, and that one wishes to use X to predict Y. In some situations, it is useful to extract a compressed representation of X that is relevant for predicting Y. This problem is formally considered by the *information bottleneck* (IB) method [1–3]. IB proposes to find a "bottleneck" variable M which maximizes prediction, formulated in terms of the mutual information $I(Y;M)$, given a constraint on compression, formulated in terms of the mutual information $I(X;M)$. Formally, this can be stated in terms of the constrained optimization problem

$$\operatorname*{arg\,max}_{M\in\Delta} I(Y;M) \quad \text{s.t.} \quad I(X;M) \le R\,, \tag{1}$$

where Δ is the set of random variables M that obey the Markov condition $Y - X - M$ [4–6]. This Markov condition states that M is conditionally independent of Y given X, and it guarantees that any information that M has about Y is extracted from X. The maximal value of $I(Y;M)$ for each possible compression value R forms what is called the *IB curve* [1].

The following example illustrates how IB might be used. Suppose that a remote weather station makes detailed recordings of meteorological data (X), which are then encoded and sent to a central server (M) and used to predict weather conditions for the next day (Y). If the channel between the weather station and server has low capacity, then the information transmitted from the weather station to the server must be compressed. Minimizing the IB objective amounts to finding a compressed representation of meteorological data which can be transmitted across a low capacity channel (have low $I(X;M)$) and used to optimally predict future weather (have high $I(Y;M)$). The IB curve specifies the trade-off between channel capacity and accurate prediction.

Numerous applications of IB exist in domains such as clustering [7,8], coding theory and quantization [9–12], speech and image recognition [13–17], and cognitive science [18]. Several recent papers have also drawn connections between IB and supervised learning, in particular, classification using neural networks [19,20]. In this context, X typically represents input vectors, Y the output classes, and M the intermediate representations used by the network, such as the activity of hidden layer(s) [21]. Existing research has considered whether intermediate representations that are optimal in the IB sense (i.e., close to the IB curve) may be better in terms of generalization error [21–23], robustness to adversarial inputs [24], detection of out-of-distribution data [25], or provide more "interesting" or "useful" intermediate representations of inputs [26]. Other related research has investigated whether stochastic gradient descent (SGD) training dynamics may drive hidden layer representations towards IB optimality [27,28].

In practice, optimal bottleneck variables are usually not found by solving the constrained optimization problem of Equation (1), but rather by finding M that maximize the so-called *IB Lagrangian* [1,6,22],

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

$\mathcal{L}_{\text{IB}}$ is the Lagrangian relaxation [29] of the constrained optimization problem of Equation (1), and β is a Lagrange multiplier that enforces the constraint $I(X;M) \le R$. In practice, $\beta \in [0,1]$ serves as a parameter that controls the trade-off between compression and prediction. As $\beta \to 1$, IB will favor maximal compression of X; for $\beta = 1$ (or any $\beta \ge 1$) the optimal M will satisfy $I(X;M) = I(Y;M) = 0$. As $\beta \to 0$, IB will favor prediction of Y; for $\beta = 0$ (or any $\beta \le 0$), there is no penalty on $I(X;M)$ and the optimal M will satisfy $I(Y;M) = I(X;Y)$, the maximum possible. It is typically easier to optimize $\mathcal{L}_{\text{IB}}$ than Equation (1), since the latter involves a complicated non-linear constraint. For this reason, optimizing $\mathcal{L}_{\text{IB}}$ has become standard in the IB literature [1,6,19,20,22,24,30,31].

However, in recent work [32] we showed that whenever Y is a deterministic function of X (or close to being one), optimizing $\mathcal{L}_{\text{IB}}$ is not longer equivalent to optimizing Equation (1). In fact, when Y is a deterministic function of X, the same M will optimize $\mathcal{L}_{\text{IB}}$ for all values of β, meaning that the IB curve cannot be explored by optimizing $\mathcal{L}_{\text{IB}}$ while sweeping β. This is a serious issue in supervised learning scenarios (as well as some other domains), where it is very common for the output Y to be a deterministic function of the input X. Nonetheless, the IB curve can still be explored by optimizing the following simple modification of the IB Lagrangian, which we called the *squared-IB Lagrangian* [32],

$$\mathcal{L}_{\text{sqIB}}(M) := I(Y;M) - \beta I(X;M)^2 \tag{3}$$

where $\beta \ge 0$ is again a parameter that controls the trade-off between compression and prediction. Unlike the case for $\mathcal{L}_{\text{IB}}$, there is always a one-to-one correspondence between M that optimize $\mathcal{L}_{\text{sqIB}}$ and solutions to Equation (1), regardless of the relationship between X and Y. In the language of optimization theory, the squared-IB Lagrangian is a "scalarization" of the multi-objective problem $\{\min I(X;M), \max I(Y;M)\}$ [33]. Importantly, unlike $\mathcal{L}_{\text{IB}}$, there can be non-trivial optimizers of $\mathcal{L}_{\text{sqIB}}$ even for $\beta \ge 1$; the relationship between β and corresponding solutions on the IB curve has been analyzed in [34]. In that work, it was also shown that the objective function of Equation (3) is part of a general family of objectives $I(Y;M) - \beta F(I(X;M))$, where F is any monotonically-increasing and strictly convex function, all of which can be used to explore the IB curve.

Unfortunately, optimizing the IB Lagrangian and squared-IB Lagrangian remains a difficult problem. First, both objectives are non-convex, so there is no guarantee that a global optimum can be found. Second, finding even a local optimum requires evaluating the mutual information terms $I(X;M)$ and $I(Y;M)$, which can involve intractable integrals. For this reason, until recently IB has been mainly developed for two limited cases. The first case is where X and Y are discrete-valued and have a small number of possible outcomes [1]. There, one can explicitly represent the full *encoding map* (the condition probability distribution of M given X) during optimization, and the relevant integrals become tractable finite sums. The second case is when X and Y are continuous-valued and jointly Gaussian. Here, the IB optimization problem can be solved analytically, and the resulting encoding and decoding maps are linear [31].

In this work, we propose a method for performing IB in much more general settings, which we call *nonlinear information bottleneck*, or *nonlinear IB* for short. Our method assumes that M is a continuous-valued random variable, but X and Y can be either discrete-valued (possibly with many states) or continuous-valued, and with any desired joint distribution. Furthermore, as suggested by the term nonlinear IB, the encoding and decoding maps can be nonlinear.

To carry out nonlinear IB, we derive a lower bound on $\mathcal{L}_{\text{IB}}$ (or, where appropriate, $\mathcal{L}_{\text{sqIB}}$) which can be maximized using gradient-based methods. As we describe in the next section, our approach makes use of the following techniques:

- We represent the distribution over X and Y using a finite number of data samples.
- We represent the encoding map $p(m|x)$ and the *decoding map* $p(y|m)$ as parameterized conditional distributions.
- We use a variational lower bound for the prediction term $I(Y;M)$, and non-parametric upper bound for the compression term $I(X;M)$, which we developed in earlier work [35].

Note that three recent papers have suggested other ways of optimizing the IB Lagrangian in general settings [24,36,37]. These papers use variational upper bounds on the compression term $I(X;M)$, which is different from our non-parametric upper bound. A detailed comparison is provided in Section 3. In that section, we also relate our approach to other work in machine learning.

In Section 4, we explain how to implement our approach using standard neural network techniques. We demonstrate its performance on several real-world datasets, and compare it to the recently-proposed *variational IB* method [24].

2. Proposed Approach

In the following, we use $H(\cdot)$ for Shannon entropy, $I(\cdot;\cdot)$ for mutual information [MI], $D_{\text{KL}}(\cdot\|\cdot)$ for Kullback–Leibler [KL] divergence. All information-theoretic quantities are in units of bits, and all logs are base-2. We use $\mathcal{N}(\mu,\Sigma)$ to indicate the probability density function of a multivariate Gaussian with mean μ and covariance matrix Σ. We use notation like $\mathbb{E}_{P(X)}[f(X)] = \int P(x)f(x)\,dx$ to indicate expectations, where $f(x)$ is some function and $P(x)$ some probability distribution. We use $\delta(\cdot,\cdot)$ for the Kronecker delta.

Let the input random variable X and the output random variable Y be distributed according to some joint distribution $Q(x,y)$, with marginals indicated by $Q(y)$ and $Q(x)$. We assume that we are provided with a "training dataset" $\mathcal{D} = \{(x_1,y_1),\ldots,(x_N,y_N)\}$, which contains N input–output pairs sampled IID from $Q(x,y)$. Let M indicate the bottleneck random variable, with outcomes in $\mathbb{R}^d$. In the derivations in this section, we assume that X and Y are continuous-valued, but our approach extends immediately to the discrete case (with some integrals replaced by sums).

Let the conditional probability $P_\theta(m|x)$ indicate a parameterized *encoding map* from input X to the bottleneck variable M, where θ is a vector of parameters. Given an encoding map, one can compute the MI between X and M, $I_\theta(X;M)$, using the joint distribution $Q_\theta(x,m) := P_\theta(m|x)Q(x)$. Similarly, one can compute the MI between Y and M, $I_\theta(Y;M)$, using the joint distribution

$$Q_\theta(y,m) := \int P_\theta(m|x)Q(x,y)\,dx\,. \tag{4}$$

We now consider the IB Lagrangian, Equation (2), as a function of the encoding map parameters,

$$\mathcal{L}_{\text{IB}}(\theta) := I_\theta(Y;M) - \beta I_\theta(X;M)\,. \tag{5}$$

In this parametric setting, we seek parameter values that maximize $\mathcal{L}_{\mathrm{IB}}(\theta)$. Unfortunately, this optimization problem is usually intractable due to the difficulty of computing the integrals in Equation (4) and in the MI terms of Equation (5). Nonetheless, it is possible to carry out an approximate form of IB by maximizing a tractable lower bound on $\mathcal{L}_{\mathrm{IB}}$, which we now derive.

First, consider any conditional probability $P_\phi(y|m)$ of outputs given bottleneck variable, where ϕ is a vector of parameters, which we call the *(variational) decoding map*. Given $P_\phi(y|m)$, the non-negativity of KL divergence leads to the following variational lower bound on the first MI term in Equation (5),

$$\begin{aligned} I_\theta(Y;M) &= H(Q(Y)) - H(Q_\theta(Y|M)) \\ &\geq H(Q(Y)) - H(Q_\theta(Y|M)) - D_{\mathrm{KL}}(Q_\theta(Y|M)\|P_\phi(Y|M)) \\ &= H(Q(Y)) + \mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right], \end{aligned} \tag{6}$$

where in the last line we've used the following identity,

$$-\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] = D_{\mathrm{KL}}(Q_\theta(Y|M)\|P_\phi(Y|M)) + H(Q_\theta(Y|M)). \tag{7}$$

Note that the inequality of Equation (6) holds for any choice of $P_\phi(y|m)$, and becomes an equality when $P_\phi(y|m)$ is equal to the "optimal" decoding map $Q_\theta(y|m)$ (as would be computed from Equation (4)). Moreover, the bound becomes tighter as the KL divergence between $P_\phi(y|m)$ and $Q_\theta(y|m)$ gets smaller. Below, we will maximize the RHS of Equation (6) with respect to ϕ, thereby bringing $P_\phi(y|m)$ closer to $Q_\theta(y|m)$.

It remains to upper bound the $I_\theta(X;M)$ term in Equation (5). To proceed, we first approximate the joint distribution of X and Y with the empirical distribution in the training dataset,

$$Q(x,y) \approx \frac{1}{N}\sum_i \delta(x_i,x)\delta(y_i,y). \tag{8}$$

We then assume that the encoding map is the sum of a deterministic function $f_\theta(x)$ plus Gaussian noise,

$$M = f_\theta(X) + Z, \tag{9}$$

where $(Z|X=x) \sim \mathcal{N}(f_\theta(x), \Sigma_\theta(x))$. Note that the noise covariance $\Sigma_\theta(x)$ can depend both on the parameters θ and the outcome of X (i.e., the noise can be heteroscedastic). Combining Equation (8) and Equation (9) implies that the bottleneck variable M will be distributed as a mixture of N equally-weighted Gaussian components, with component i having distribution $\mathcal{N}(f_\theta(x_i), \Sigma_\theta(x_i))$. We can then employ the following non-parametric upper bound on MI, which was derived in a recent paper [35]:

$$I_\theta(X;M) \leq \hat{I}_\theta(X;M) := -\frac{1}{N}\sum_i \log\frac{1}{N}\sum_j e^{-D_{\mathrm{KL}}\left[\mathcal{N}(f_\theta(x_i),\Sigma_\theta(x_i))\middle\|\mathcal{N}(f_\theta(x_j),\Sigma_\theta(x_j))\right]}. \tag{10}$$

(Note that the published version of [35] contains some typos which are corrected in the latest arXiv version at arxiv.org/abs/1706.02419.)

Equation (10) bounds the MI in terms of the pairwise KL divergences between the Gaussian components of the mixture distribution of M. It is useful because the KL divergence between two d-dimensional Gaussians has a closed-form expression,

$$D_{\mathrm{KL}}\left[\mathcal{N}(\mu',\Sigma')\middle\|\mathcal{N}(\mu,\Sigma)\right] = \frac{1}{2}\left[\ln\frac{\det\Sigma}{\det\Sigma'} + (\mu'-\mu)\Sigma^{-1}(\mu'-\mu) + \mathrm{tr}(\Sigma^{-1}\Sigma') - d\right]. \tag{11}$$

Furthermore, in the special case when all components have the same covariance and can be grouped into well-separated clusters, the upper bound of Equation (10) becomes tight [35]. As we will see below, this special case is a commonly encountered solution to the optimization problem considered here.

Combining Equation (6) and Equation (10) provides the following tractable lower bound for the IB Lagrangian,

$$\mathcal{L}_{\text{IB}}(\theta) \geq \hat{\mathcal{L}}_{\text{IB}}(\theta,\phi) := \mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] - \beta \hat{I}_\theta(X;M) \tag{12}$$

where we dropped the additive constant $H(Q(Y))$ (which does not depend on the parameter values and is therefore irrelevant for optimization). We refer to Equation (12) as the *nonlinear IB* objective.

As mentioned in the introduction, in cases where Y is a deterministic function of X (or close to being one), it is no longer possible to explore the IB curve by optimizing the IB Lagrangian for different values of β [19,32,34]. Nonetheless, it is always possible to explore the IB curve by instead optimizing the squared-IB Lagrangian, Equation (3). The above derivations also lead to the following tractable lower bound for the squared-IB Lagrangian,

$$\mathcal{L}_{\text{sqIB}}(\theta) \geq \hat{\mathcal{L}}_{\text{sqIB}}(\theta,\phi) := \mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] - \beta\left[\hat{I}_\theta(X;M)\right]^2. \tag{13}$$

Note that maximizing the expectation term $\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right]$ is equivalent to minimizing the usual cross-entropy loss in supervised learning. (Note that mean squared error, the typical loss function used for training regression models, can also be interpreted as a cross-entropy term [38] (pp. 132–134).) From this point of view, Equation (12) and Equation (13) can be interpreted as adding an information-theoretic regularization term to the regular objective of supervised learning.

For optimization purposes, the compression term $\hat{I}_\theta(X;M)$ can be computed from data using Equations (10) and (11), while the expectation term $\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right]$ can be estimated as $\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] \approx \frac{1}{N}\sum_i \log P_\phi(y_i|m_i)$, where m_i indicates samples from $P_\theta(m|x_i)$. Assuming that f_θ is differentiable with respect to θ and P_ϕ is differentiable with respect to ϕ, the optimal θ and ϕ can be selected by using gradient-based methods to maximize Equation (12) or Equation (13), as desired. In practice, this optimization will typically be done using stochastic gradient descent (SGD), i.e., by computing the gradient using randomly sampled mini-batches rather than the whole training dataset. In fact, mini-batching becomes necessary for large datasets, since evaluating $\hat{I}_\theta(X;M)$ involves $O(n^2)$ operations, where n is the number of data points in the batch used to compute the gradient, which becomes prohibitively slow for very large n. At the same time, $\hat{I}_\theta(X;M)$ is closely related to kernel-density estimators [35], and it is known that the number of samples required for accurate kernel-density estimates grows rapidly as dimensionality increases [39]. Thus, mini-batches should not be too small when d (the dimensionality of the bottleneck variable) is large. In some cases, it may be useful to estimate the gradient of $\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right]$ and the gradient of $\hat{I}_\theta(X;M)$ using mini-batches of different sizes. More implementation details are discussed below in Section 4.1.

Note that the approach described here is somewhat different (and simpler) than in previous versions of this manuscript [40,41]. In previous versions, we represented the marginal distribution $Q(x)$ with a mixture of Gaussians, rather than with the empirical distribution in the training data. However, we found that this increased complexity but was not necessary for good performance. Furthermore, we previously focused only on optimizing a bound on the IB Lagrangian, Equation (12). In subsequent work [32], we showed that the IB Lagrangian is inadequate for many supervised learning scenarios, including some of those explored in Section 4.2, and that the squared-IB Lagrangian should be used instead. In this work, we report performance when optimizing Equation (13), a bound on the squared-IB Lagrangian.

3. Relation to Prior Work

In this section, we relate our proposed method to prior work in machine learning.

3.1. Variational IB

Recently, there have been three other proposals for performing IB for continuous and possibly non-Gaussian random variables using neural networks [24,36,37], the most popular of which is called *variational IB* (VIB) [24]. As in our approach, these methods propose tractable lower bounds on the $\mathcal{L}_{\mathrm{IB}}$ objective. They employ the same variational bound for the prediction MI term $I(Y;M)$ as our Equation (6). These methods differ from ours, however, in how they bound the compression term, $I_\theta(X;M)$. In particular, they all use some form of the following variational upper bound,

$$I_\theta(X;M) = D_{\mathrm{KL}}(P_\theta(M|X)\|R(M)) - D_{\mathrm{KL}}(P_\theta(M)\|R(M)) \leq D_{\mathrm{KL}}(P_\theta(M|X)\|R(M)) \,, \tag{14}$$

where R is some surrogate marginal distribution over the bottleneck variable M. Combining with Equation (6) leads to the following variational lower bound for $\mathcal{L}_{\mathrm{IB}}$,

$$\mathcal{L}_{\mathrm{IB}}(M) \geq \mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] - \beta D_{\mathrm{KL}}(P_\theta(M|X)\|R(M)) + \mathrm{const}\,. \tag{15}$$

The three aforementioned papers differ in how they define the surrogate marginal distribution R. In [24], R is a standard multivariate normal distribution, $\mathcal{N}(0,\mathbf{I})$. In [36], R is a product of Student's t-distributions. The scale and shape parameters of each t-distribution are optimized during training, in this way tightening the bound in Equation (14). In [37], two surrogate distributions are considered, the improper log-uniform and the log-normal, with the appropriate choice depending on the particular activation function (non-linearity) used in the neural network.

In addition, the encoding map $P_\theta(m|x)$ in [36] and [24] is a deterministic function plus Gaussian noise, same as in Equation (9). In [37], the encoding map consists of a deterministic function with multiplicative, rather than additive, noise.

These alternative methods have potential advantages and disadvantages compared to our approach. On one hand, they are more computationally efficient: Our non-parametric estimator of $\hat{I}_\theta(X;M)$ requires $O(n^2)$ operations per mini-batch (where n is the size of the mini-batch), while the variational bound of Equation (14) requires $O(n)$ operations. On the other hand, our non-parametric estimator is expected to give a better estimate of the true MI $I(X;M)$ [35]. We provide a comparison between our approach and variational IB [25] in Section 4.2.

3.2. Neural Networks and Kernel Density Entropy Estimates

A key component of our approach is using a differentiable upper bound on MI, $\hat{I}_\theta(X;M)$. As discussed in [35], this bound is related to non-parametric kernel-density estimators of MI. See [42–46] for related work on using neural networks to optimize non-parametric estimates of information-theoretic functions. This technique can also be related to kernel-based estimation of the likelihood of held-out data for neural networks (e.g., [47]). In these later approaches, however, the likelihood of held-out data is estimated only once, as a diagnostic measure once learning is complete. We instead propose to train the network by directly incorporating our non-parametric estimator $\hat{I}_\theta(X;M)$ in the objective function.

3.3. Auto-Encoders

Auto-encoders are unsupervised learning architectures that learn to reconstruct a copy of the input X, while using some intermediate representations (such as a hidden layer in a neural network). Auto-encoders have some conceptual relationships to IB, in that the intermediate representations are sometimes restricted in terms of dimensionality, or with information-theoretic penalties on hidden layer coding length [48,49]. Similar penalties have also been explored in a supervised learning scenario in [50]. In that work, however, hidden layer states were treated as discrete-valued, limiting the flexibility and information capacity of hidden representations.

More recently, *denoising auto-encoders* [51] have attracted attention. Denoising auto-encoders constrain the amount of information passing from input to hidden layers by injecting noise into the hidden layer activity, similarly to our noisy mapping from the input to the bottleneck layer. Previous work on auto-encoders has considered either penalizing hidden layer coding length *or* injecting noise into the map, rather than combing the two as we do here. Moreover, denoising auto-encoders do not have a notion of an "optimal" noise level, since less noise will always improve prediction error on the training data. Thus, they cannot directly adapt the noise level (as done in our method).

Finally, *variational auto-encoders* [52] [VAEs] are recently-proposed architectures which learn generative models from unsupervised data (i.e., after training, they can be used to generate new samples that resemble training data). Interestingly, the objective optimized in VAE training, called "ELBO", contains both a prediction term and a compression term and can be seen as a special case of the variational IB objective [24,37,53,54]. In principle, it may be fruitful to replace the compression term in the ELBO with our MI estimator $\hat{I}_\theta(X;M)$. Given our reported performance below, this may result in better compression, though it might also complicate sampling from the latent variable space. We leave this line of research for future work.

4. Experiments

In this section, we first explain how to implement nonlinear IB using neural network techniques. We then evaluate its on several datasets, and compare it to the variational IB (VIB) method. We demonstrate that, compared to VIB, nonlinear IB achieves better performance and uncovers different kinds of representations.

4.1. Implementation

Any implementation of nonlinear IB requires a way to compute the encoding map $P_\theta(m|x)$ and decoding map $P_\phi(y|m)$, as well as a way to choose the parameters of these maps so as to maximize the nonlinear IB objective. Here we explain how this can be done using standard neural network methods.

The encoding map $P_\theta(m|x)$, Equation (9), is implemented in the following way: First, several neural network layers with parameters θ implement the (possibly nonlinear) deterministic function $f_\theta(x)$. The output of these layers is then added to zero-centered Gaussian noise with covariance $\Sigma_\theta(x)$, which becomes the state of the *bottleneck layer*. This is typically done via the "reparameterization trick" [52], in which samples of Gaussian noise are passed through several deterministic layers (whose parameters are also indicated by θ) and then added to $f_\theta(x)$. Note that due to the presence of noise, the neural network is stochastic: even with parameters held constant, different states of the bottleneck layer are sampled during different NN evaluations. This stochasticity guarantees that the mutual information $I(X;M)$ is finite [26,28].

In all of the experiments described below, the encoding map consists of two layers with 128 ReLU neurons each, following by a layer of 5 linear neurons. In addition, for simplicity we use a simple homoscedastic noise model: $\Sigma_\theta(x) = \sigma^2\mathbf{I}$, where σ^2 is a parameter the sets the scale of the noise variance. This noise model permits us to rewrite the MI bound of Equation (10) in terms of the following simple expression,

$$\hat{I}_\theta(X;M) = -\frac{1}{N}\sum_i \log \frac{1}{N}\sum_j e^{-\frac{1}{2\sigma^2}\|f_\theta(x_i)-f_\theta(x_j)\|_2^2}. \tag{16}$$

For purposes of comparison, we use this same homoscedastic noise model for both nonlinear IB and for VIB (note that the original VIB paper [24] used a heteroscedastic noise model; investigating the performance of nonlinear IB with heteroscedastic noise remains for future work).

In our runs, the noise parameter σ^2 was one of the trainable parameters in θ. The initial value of σ^2 should be chosen with some care. If the initial σ^2 is too small, the Gaussian components that make up the mixture distribution of M will be many standard deviations away from each other and $\hat{I}_\theta(X;M)$ (as well as $I(X;M)$) will be exponentially close to the constant $\log N$ [35]. In this case, the gradient of the compression term $\hat{I}_\theta(X;M)$ with respect to θ will also be exponentially small, and the optimizer will not be able to learn to compress. On the other hand, when σ^2 is too large, the resulting noise can swamp gradient information arising from the accuracy (cross-entropy) term, cause the optimizer to collapse to a "trivial" maximally-compressed model in which $I(X;M) \approx I(Y;M) \approx 0$. Nonetheless, the optimization is robust to several orders of magnitude of variation of the initial value of σ^2. In the experiments below, we uses the initial value $\sigma^2 = 1$, which works sufficiently well in practice. (Note that the scale of the noise can also be trained by changing the parameters of the 5-neuron linear layer; thus, in our neural network architecture, having a trainable σ^2 is not strictly necessary.)

To implement the decoding map $P_\phi(y|m)$, the bottleneck layer states are passed through several deterministic neural network layers with parameters ϕ. In the experiments described below, the decoding map is implemented with a single layer with 128 ReLU neurons, followed by a linear output layer. The log decoding probability ($\log P_\phi(y|m)$) is then evaluated using the network output and an appropriately-chosen cost function: cross-entropy loss of the softmax of the output for classification, and mean squared error (MSE) of the output for regression.

In the experiments below, we use nonlinear IB to optimize the bound on the "squared-IB Lagrangian", Equation (13), rather than the bound on the IB Lagrangian, Equation (12). For comparison purposes, we also optimize the following "squared" version of the VIB objective, Equation (15),

$$\mathcal{L}_{\text{sq-VIB}} := \mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right] - \beta\left[D_{\text{KL}}(P_\theta(M|X)\|R(M))\right]^2. \tag{17}$$

As in the original VIB paper, we take $R(m)$ to be the standard Gaussian $\mathcal{N}(0,\mathbf{I})$. We found that optimizing the squared-IB bounds, Equation (13) and Equation (17), produced quantitatively similar results to optimizing Equation (12) and Equation (15), but was more numerically robust when exploring the full range of the IB curve. For an explanation of why this occurs, see the discussion and analysis in [32]. We report performance of nonlinear IB and VIB when optimizing bounds on the IB Lagrangian, Equation (12) and Equation (15), in the Supplementary Material.

We use the Adam [55] optimizer with standard TensorFlow settings and mini-batches of size 256. To avoid over-fitting, we use early stopping: we split the training data into 80% actual training data and 20% validation data; training is stopped once the objective on the validation dataset did not improve for 50 epochs.

A TensorFlow implementation of our approach is provided at `https://github.com/artemyk/nonlinearIB`. An independent PyTorch implementation is available at `https://github.com/burklight/nonlinear-IB-PyTorch`.

4.2. Results

We report the performance of nonlinear IB on two different classification datasets (MNIST and FashionMNIST) and one regression dataset (California housing prices). We also compare it with the recently-proposed variational IB (VIB) method [24]. Here we focus purely on the ability of these methods to optimize the IB objective on training and testing data. We leave for future work comparisons of these methods in terms of adversarial robustness [24], detection of out-of-distribution data [25], and other desirable characteristics that may emerge from IB training.

We optimize both the nonlinear IB (Equation (13)) and the VIB (Equation (17)) objectives for different values of β, producing a series of models that explore the trade-off between compression and prediction. We vary $\beta \in [10^{-3}, 2]$ for classification tasks and $\beta \in [10^{-5}, 2]$ for the regression task. These ranges were chosen empirically so that the resulting models fully explore the IB curve.

To report our results, we use *information plane* (info-plane) diagrams [27], which visualize the performance of different models in terms of the compression term ($I(X;M)$, the x-axis) and the prediction term ($I(Y;M)$, the y-axis) both on training and testing data. For the info-plane diagrams, we use Monte Carlo sampling to get an accurate estimate of $I(X;M)$ terms. To estimate the $I(Y;M) = H(Y) - H(Y|M)$ term, we use two different approaches. For classification datasets, we approximate $H(Y)$ using the empirical entropy of the class labels in the dataset, and approximate the conditional entropy with the cross-entropy loss, $H(Y|M) \approx -\mathbb{E}_{Q_\theta(Y,M)}\left[\log P_\phi(Y|M)\right]$. Note that the resulting MI estimate is an additive constant away from the cross-entropy loss. For the regression dataset, we approximate $H(Y)$ via the entropy of a Gaussian with variance Var(Y), and approximate $H(Y|M)$ via the entropy of a Gaussian with variance equal to the mean-squared-error. This results in the estimate $I(Y;M) \approx \frac{1}{2}\log(\mathrm{Var}(Y)/\mathrm{MSE})$. Finally, we also use scatter plots to visualize the activity of the hidden layer for models trained with different objectives.

We first consider the *MNIST* dataset of hand-drawn digits, which contains 60,000 training images and 10,000 testing images. Each image is 28-by-28 pixels (784 total pixels, so $X \in \mathbb{R}^{784}$), and is classified into 1 of 10 classes corresponding to the digit identity ($Y \in \{1,\ldots,10\}$).

The top row of Figure 1 shows $I(Y;M)$ and $I(X;M)$ values achieved by nonlinear IB and VIB on the MNIST dataset. As can be seen, nonlinear IB achieves better prediction values at the same level of compression than VIB, both on training and testing data. The difference is especially marked near the "corner point" $I(X;M) = I(Y;M) \approx \log 10$ (which corresponds to maximal compression, given perfect prediction), where nonlinear IB achieved ≈ 0.1 bits better prediction at the same compression level (see also Table 1).

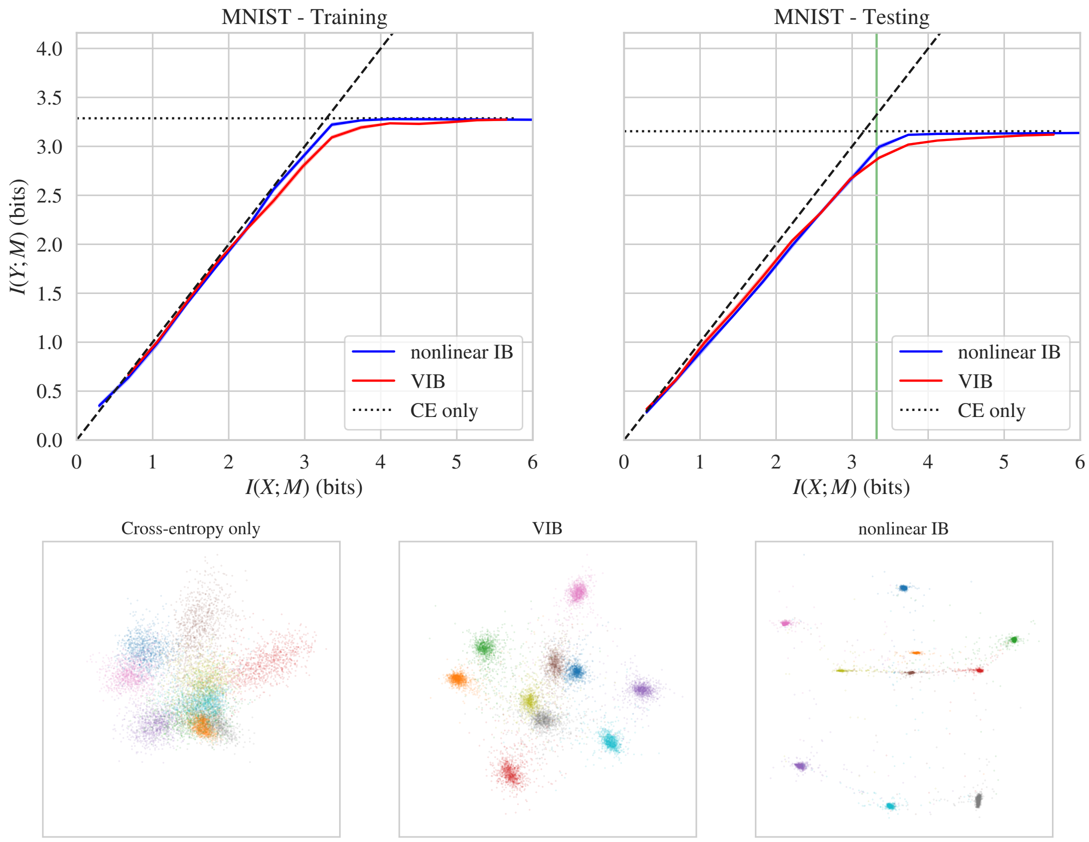


Figure 1. **Top row**: Info-plane diagrams for nonlinear IB and variational IB (VIB) on the MNIST training (**left**) and testing (**right**) data. The solid lines indicate means across five runs, shaded region indicates the standard error of the mean. The black dashed line is the data-processing inequality bound $I(Y;M) \leq I(X;M)$, the black dotted line indicates the value of $I(Y;M)$ achieved by a baseline model trained only to optimize cross-entropy. **Bottom row**: Principal component analysis (PCA) projection of bottleneck layer activity (on testing data, no noise) for models trained with regular cross-entropy loss (**left**), VIB (**middle**), and nonlinear IB (**right**) objectives. The location of the nonlinear IB and VIB models shown in the bottom row is indicated with the green vertical line in the top right panel.

Further insight is provided by considering the bottleneck representations found when training with nonlinear IB versus VIB versus regular cross-entropy loss. To visualize these bottleneck representations, we selected three models: a baseline model trained only to optimize cross-entropy loss, a model trained with nonlinear IB, and a model trained with VIB (the latter two models were chosen to both have $I(X;M) \approx \log 10$). We then measured the activity of their 5-neuron bottleneck hidden layer on the testing dataset, projected down to two dimensions using principal component analysis (PCA). Figure 1 visualizes these two-dimensional projections for these three models, with colors indicating class label (digit identity). Training with VIB and nonlinear IB objectives causes inputs corresponding to different digits to fall into well-separated clusters, unlike training with cross-entropy loss. Moreover, the clustering is particularly tight for nonlinear IB, meaning that the bottleneck states carry almost no information about input vectors beyond class identity. Note that in this regime, where Gaussian components are grouped into tightly separate clusters, our MI upper bound $\hat{I}_\theta(X;M)$ becomes exact [35].

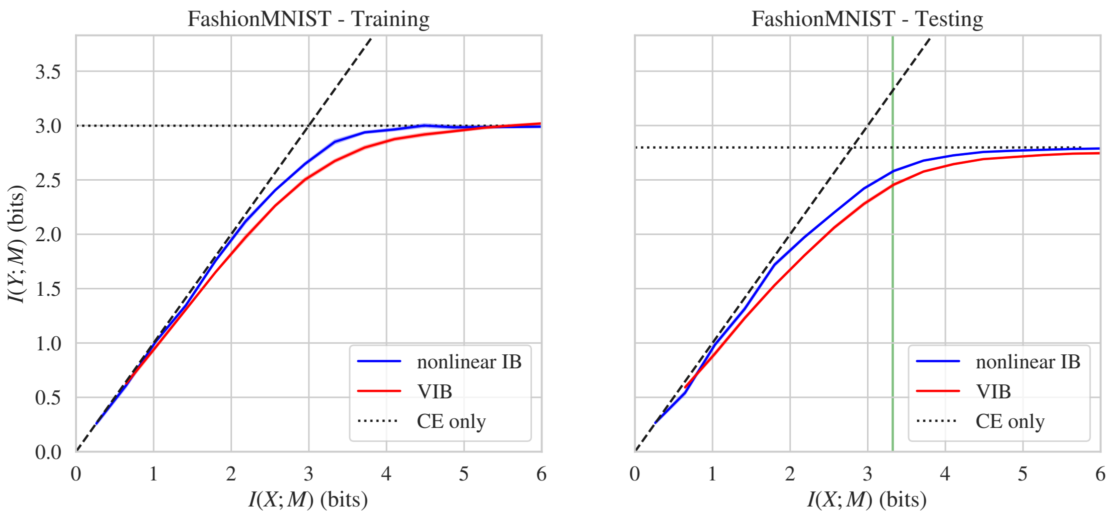
Table 1. Amount of prediction $I(Y;M)$ achieved at compression level $I(X;M) = \log 10$ for both nonlinear IB and VIB.

Dataset		Nonlinear IB	VIB
MNIST	Training	**3.22**	3.09
	Testing	**2.99**	2.88
FashionMNIST	Training	**2.85**	2.67
	Testing	**2.58**	2.46
California housing	Training	**1.37**	1.26
	Testing	**1.13**	1.07

In the next experiment, we considered the recently-proposed *FashionMNIST* dataset. FashionMNIST has the same structure as the MNIST dataset (28 × 28 images grouped into 10 classes, with 60,000 training and 10,000 testing images). Instead of hand-written digits, however, FashionMNIST includes images of clothes labeled with classes such as "Dress", "Coat", and "Sneaker". This dataset was designed as a drop-in replacement for MNIST which addresses the problem that MNIST is too easy for modern machine learning (e.g., it is fairly straightforward to achieve ≈99% test accuracy on MNIST) [56]. FashionMNIST is a more difficult dataset, with typical test accuracies of ≈90%–95%.

The top row Figure 2 shows $I(Y;M)$ and $I(X;M)$ values achieved by nonlinear IB and VIB on the FashionMNIST dataset. Compared to VIB, nonlinear IB again achieves better prediction values at the same level of compression, both on training and testing data. The difficulty of FashionMNIST is evident in the fact that neither method gets very close to the corner point $I(X;M) = I(Y;M) \approx \log 10$. Nonetheless, nonlinear IB performed better than VIB at a range of compression values, often extracting ≈ 0.15 additional bits of prediction at the same compression level (see also Table 1).

As for MNIST, we consider the bottleneck representations uncovered when training on FashionMNIST with cross-entropy loss only versus nonlinear IB versus VIB (the latter two models were chosen to have $I(X;M) \approx \log 10$). We measured the activity of the 5-neuron bottleneck layer on the testing dataset, projected down to two dimensions using PCA. The bottom row of Figure 2 visualizes these two-dimensional projections for these three models, with colors indicating class label (digit identity). It can again be seen that models trained with VIB and nonlinear IB map inputs into separated clusters, but that the clusters are significantly tighter for nonlinear IB.



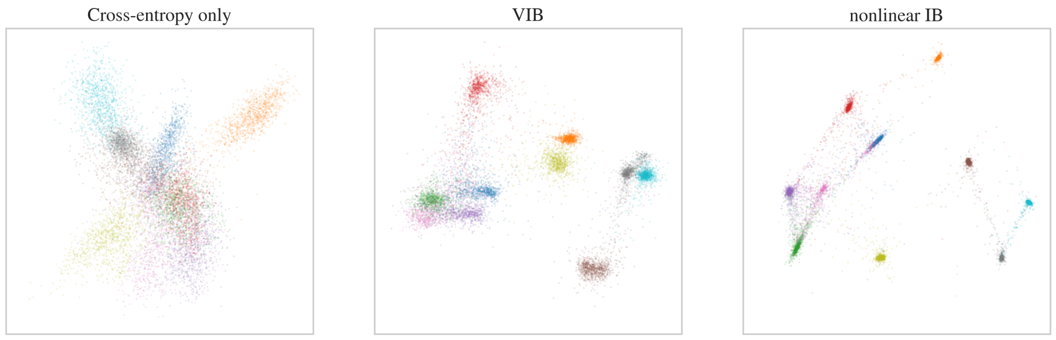


Figure 2. Top row: Info-plane diagrams for nonlinear IB and VIB on the FashionMNIST dataset. **Bottom row**: PCA projection of bottleneck layer activations for models trained only to optimize cross-entropy (**left**), VIB (**middle**), and nonlinear IB (**right**) objectives. See caption of Figure 1 for details.

In our final experiment, we considered the *California housing prices* dataset. This is a regression dataset based on the 1990 California census, originally published in [57] (we use the version distributed with the `scikit-learn` package [58]). It consists of $N = 20,640$ total samples, with one dependent variable (the house price) and 8 independent variables (such as "longitude", "latitude", and "number of rooms"). We used the log-transformed house price as the dependent variable Y (this made the distribution of Y closer to a Gaussian). To prepare the training and testing data, we first dropped 992 samples in which the house price was equal to or greater than \$500,000 (prices were clipped at this upper value in the dataset, which distorted the distribution of the dependent variable). We then randomly split the remaining samples into an 80% training and 20% testing dataset (the training dataset was then further split into the actual training dataset and a validation dataset, see above).

The top row of Figure 3 shows $I(Y;M)$ and $I(X;M)$ values achieved by nonlinear IB and VIB on the California housing prices dataset. Nonlinear IB achieves better prediction values at the same level of compression than VIB, both on training and testing data (see also Table 1). As for the other datasets, we also show the bottleneck representations uncovered when training on California housing prices dataset with MSE loss only versus nonlinear IB versus VIB (the latter two models were chosen to have $I(X;M) \approx \log 10$). The bottom row of Figure 3 visualizes the two-dimensional PCA projections of bottleneck layer activity for these three models, with colors indicating the dependent variable (log housing price). The bottleneck representations uncovered when training with MSE loss only and when training with VIB were somewhat similar. Nonlinear IB, however, finds a different and almost perfectly one-dimensional bottleneck representation. In fact, for the nonlinear IB model, the first principal component explains 99.8% of the variance in bottleneck layer activity on testing data. For the models trained with MSE loss and VIB, the first principal component explains only 76.6% and 69% of the variance, respectively. The one-dimensional representation uncovered by nonlinear IB compresses away all information about the input vectors which is not relevant for predicting the dependent variable.

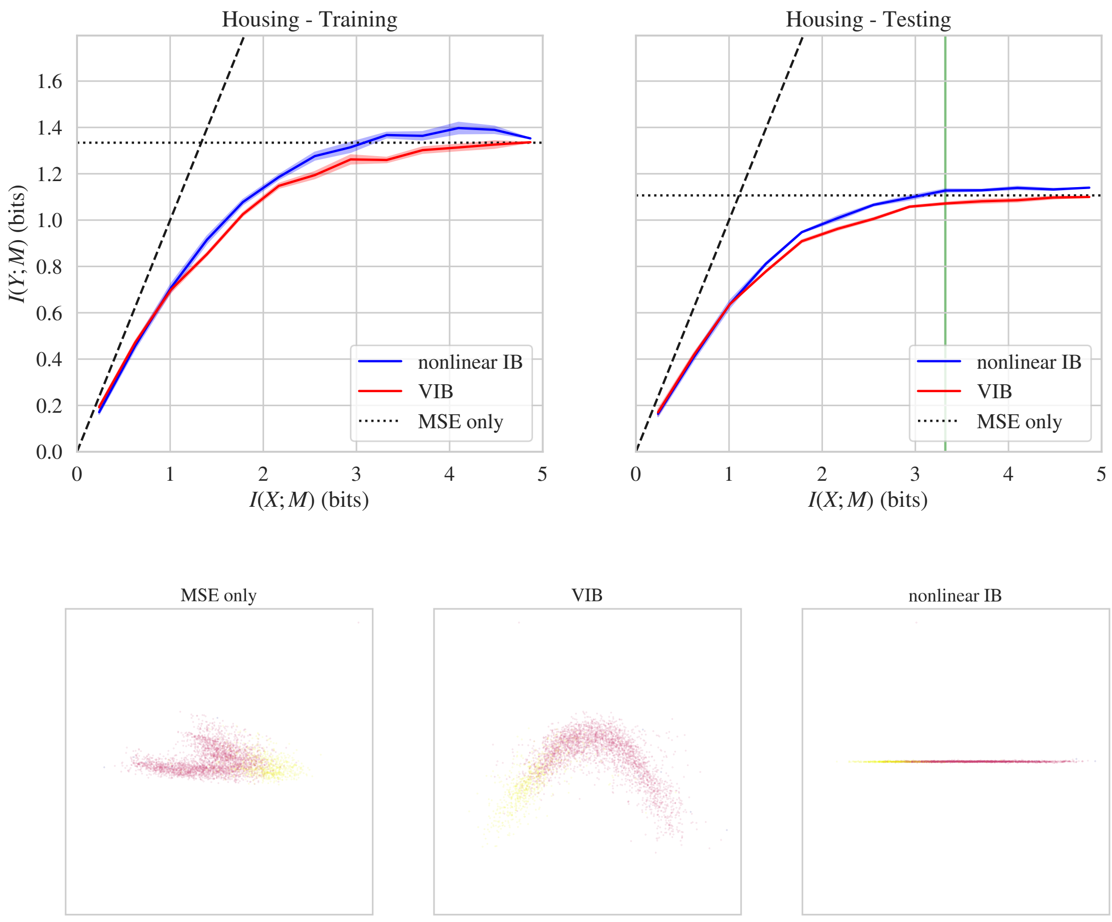


Figure 3. **Top row**: Information plane diagrams for nonlinear IB and VIB on the California housing prices dataset. **Bottom row**: PCA projection of bottleneck layer activations for models trained only to optimize mean squared error (MSE) (**left**), VIB (**middle**), and nonlinear IB (**right**) objectives. See caption of Figure 1 for details.

We finish by presenting some of our numerical results in Table 1. In particular, we quantify the amount of prediction, $I(Y; M)$, achieved when training with nonlinear IB and VIB at the compression level $I(X; M) = \log 10$, for training and testing datasets of the three datasets considered above. Nonlinear IB consistently achieves better prediction at a fixed level of compression.

5. Conclusions

We propose "nonlinear IB", a method for exploring the information bottleneck [IB] trade-off curve in a general setting. We allow the input and output variables to be discrete or continuous (though we assume a continuous bottleneck variable). We also allow for arbitrary (e.g., non-Gaussian) joint distributions over inputs and outputs and for non-linear encoding and decoding maps. We gain this generality by exploiting a new tractable and differentiable bound on the IB objective.

We describe how to implement our method using off-the-shelf neural network software, and apply it to several standard classification and regression problems. We find that nonlinear IB is able to effectively discover the tradeoff curve, and find solutions that are superior compared with competing methods. We also find that the intermediate representations discovered by nonlinear IB have visibly tighter clusters in the classification problems. In the regression problem, nonlinear IB discovers a one-dimensional intermediate representation.

We have successfully demonstrated the ability of nonlinear IB to explore the IB curve. It is possible that increased compression may lead to other benefits in supervised learning, such as improved generalization performance or increased robustness to adversarial inputs. Exploring its efficacy in these domains remains for future work.

Supplementary Materials: The following are available online at http://www.mdpi.com/1099-4300/21/12/1181/s1: Figure S1: Performance of nonlinear IB and VIB when optimizing bounds on regular IB objective.

Author Contributions: Conceptualization, A.K.; Funding acquisition, D.H.W.; Software, A.K. and B.D.T.; Visualization, A.K.; Writing—original draft, A.K.; Writing—review & editing, A.K., B.D.T. and D.H.W.

Funding: This research was funded by National Science Foundation: CHE-1648973; Foundational Questions Institute: FQXi-RFP-1622; Air Force Office of Scientific Research: A9550-15-1-0038.

Acknowledgments: We thank Steven Van Kuyk and Borja Rodríguez Gálvez for helpful comments. We would also like to thank the Santa Fe Institute for helping to support this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tishby, N.; Pereira, F.; Bialek, W. The information bottleneck method. In Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, USA, 22–24 September 1999.
2. Dimitrov, A.G.; Miller, J.P. Neural coding and decoding: Communication channels and quantization. *Netw. Comput. Neural Syst.* **2001**, *12*, 441–472. [CrossRef]
3. Samengo, I. Information loss in an optimal maximum likelihood decoding. *Neural Comput.* **2002**, *14*, 771–779. [CrossRef]
4. Witsenhausen, H.; Wyner, A. A conditional entropy bound for a pair of discrete random variables. *IEEE Trans. Inf. Theory* **1975**, *21*, 493–501. [CrossRef]
5. Ahlswede, R.; Körner, J. Source Coding with Side Information and a Converse for Degraded Broadcast Channels. *IEEE Trans. Inf. Theory* **1975**, *21*, 629–637. [CrossRef]
6. Gilad-Bachrach, R.; Navot, A.; Tishby, N. An Information Theoretic Tradeoff between Complexity and Accuracy. In *Learning Theory and Kernel Machines*; Goos, G., Hartmanis, J., van Leeuwen, J., Schölkopf, B., Warmuth, M.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2777, pp. 595–609.
7. Slonim, N.; Tishby, N. Document clustering using word clusters via the information bottleneck method. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, 24–28 July 2000; pp. 208–215.
8. Tishby, N.; Slonim, N. Data clustering by markovian relaxation and the information bottleneck method. In *Advances in Neural Information Processing Systems 13 (NIPS 2000)*; MIT Press: Cambridge, MA, USA, 2001; pp. 640–646.
9. Cardinal, J. Compression of side information. In Proceedings of the 2003 International Conference on Multimedia and Expo, Baltimore, MD, USA, 6–9 July 2003; pp. 569–572.
10. Zeitler, G.; Koetter, R.; Bauch, G.; Widmer, J. Design of network coding functions in multihop relay networks. In Proceedings of the 2008 5th International Symposium on Turbo Codes and Related Topics, Lausanne, Switzerland, 1–5 September 2008; pp. 249–254.
11. Courtade, T.A.; Wesel, R.D. Multiterminal source coding with an entropy-based distortion measure. In Proceedings of the 2011 IEEE International Symposium on Information Theory, St. Petersburg, Russia, 31 July–5 August 2011; pp. 2040–2044.
12. Lazebnik, S.; Raginsky, M. Supervised learning of quantizer codebooks by information loss minimization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 1294–1309. [CrossRef] [PubMed]
13. Winn, J.; Criminisi, A.; Minka, T. Object categorization by learned universal visual dictionary. In Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Beijing, China, 17–21 October 2005; Volume 2, pp. 1800–1807.

14. Hecht, R.M.; Noor, E.; Tishby, N. Speaker recognition by Gaussian information bottleneck. In Proceedings of the 10th Annual Conference of the International Speech Communication Association, Brighton, UK, 6–10 September 2009.
15. Yaman, S.; Pelecanos, J.; Sarikaya, R. Bottleneck features for speaker recognition. In Proceedings of the Speaker and Language Recognition Workshop, Singapore, 25–28 June 2012.
16. Van Kuyk, S.; Kleijn, W.B.; Hendriks, R.C. On the information rate of speech communication. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 5625–5629.
17. Van Kuyk, S. Speech Communication from an Information Theoretical Perspective. Ph.D. Thesis, Victoria University of Wellington, Wellington, New Zealand, 2019.
18. Zaslavsky, N.; Kemp, C.; Regier, T.; Tishby, N. Efficient compression in color naming and its evolution. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 7937–7942. [CrossRef] [PubMed]
19. Rodríguez Gálvez, B. The Information Bottleneck: Connections to Other Problems, Learning and Exploration of the IB Curve. Master's Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, June 2019.
20. Hafez-Kolahi, H.; Kasaei, S. Information Bottleneck and its Applications in Deep Learning. *arXiv* **2019**, arXiv:1904.03743.
21. Tishby, N.; Zaslavsky, N. Deep learning and the information bottleneck principle. In Proceedings of the 2015 IEEE Information Theory Workshop (ITW), Jerusalem, Israel, 26 April–1 May 2015; pp. 1–5.
22. Shamir, O.; Sabato, S.; Tishby, N. Learning and generalization with the information bottleneck. *Theor. Comput. Sci.* **2010**, *411*, 2696–2711. [CrossRef]
23. Vera, M.; Piantanida, P.; Vega, L.R. The Role of the Information Bottleneck in Representation Learning. In Proceedings of the 2018 IEEE International Symposium on Information Theory (ISIT), Vail, CO, USA, 17–22 June 2018; pp. 1580–1584.
24. Alemi, A.A.; Fischer, I.; Dillon, J.V.; Murphy, K. Deep Variational Information Bottleneck. In Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
25. Alemi, A.A.; Fischer, I.; Dillon, J.V. Uncertainty in the variational information bottleneck. *arXiv* **2018**, arXiv:1807.00906.
26. Amjad, R.A.; Geiger, B.C. Learning Representations for Neural Network-Based Classification Using the Information Bottleneck Principle. *arXiv* **2018**, arXiv:1802.09766.
27. Shwartz-Ziv, R.; Tishby, N. Opening the Black Box of Deep Neural Networks via Information. *arXiv* **2017**, arXiv:1703.00810.
28. Saxe, A.; Bansal, Y.; Dapello, J.; Advani, M.; Kolchinsky, A.; Tracey, B.; Cox, D. On the information bottleneck theory of deep learning. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
29. Lemaréchal, C. Lagrangian relaxation. In *Computational Combinatorial Optimization*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 112–156.
30. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*; John Wiley & Sons: Hoboken, NJ, USA, 2012.
31. Chechik, G.; Globerson, A.; Tishby, N.; Weiss, Y. Information bottleneck for Gaussian variables. *J. Mach. Learn. Res.* **2005**, *6*, 165–188.
32. Kolchinsky, A.; Tracey, B.D.; Van Kuyk, S. Caveats for information bottleneck in deterministic scenarios. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
33. Miettinen, K. *Nonlinear Multiobjective Optimization*; Springer: Boston, MA, USA, 1998. [CrossRef]
34. Rodríguez Gálvez, B.; Thobaben, R.; Skoglund, M. The Convex Information Bottleneck Lagrangian. *arXiv* **2019**, arXiv:1911.11000.
35. Kolchinsky, A.; Tracey, B.D. Estimating Mixture Entropy with Pairwise Distances. *Entropy* **2017**, *19*, 361. [CrossRef]
36. Chalk, M.; Marre, O.; Tkacik, G. Relevant sparse codes with variational information bottleneck. In Proceedings of the 2016 Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016; pp. 1957–1965.
37. Achille, A.; Soatto, S. Information Dropout: Learning optimal representations through noise. *arXiv* **2016**, arXiv:1611.01353.
38. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.

39. Silverman, B.W. *Density Estimation for Statistics and Data Analysis*; Routledge: New York, NY, USA, 2018.
40. Kolchinsky, A.; Wolpert, D.H. Supervised learning with information penalties. In Proceedings of the 2016 Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016.
41. Kolchinsky, A.; Tracey, B.D.; Wolpert, D.H. Nonlinear Information Bottleneck. *Entropy* **2019**, *21*, 1181. [CrossRef]
42. Schraudolph, N.N. Optimization of entropy with neural networks. Ph.D. Thesis, University of California, San Diego, CA, USA, 1995.
43. Schraudolph, N.N. Gradient-based manipulation of nonparametric entropy estimates. *IEEE Trans. Neural Netw.* **2004**, *15*, 828–837. [CrossRef]
44. Shwartz, S.; Zibulevsky, M.; Schechner, Y.Y. Fast kernel entropy estimation and optimization. *Signal Process.* **2005**, *85*, 1045–1058. [CrossRef]
45. Torkkola, K. Feature extraction by non-parametric mutual information maximization. *J. Mach. Learn. Res.* **2003**, *3*, 1415–1438.
46. Hlavávcková-Schindler, K.; Palus, M.; Vejmelka, M.; Bhattacharya, J. Causality detection based on information-theoretic approaches in time series analysis. *Phys. Rep.* **2007**, *441*, 1–46. [CrossRef]
47. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 2014 Conference on Neural Information Processing Systems (NIPS 2014), Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
48. Hinton, G.E.; Zemel, R.S. Autoencoders, minimum description length, and Helmholtz free energy. In *Advances in Neural Information Processing Systems 7 (NIPS 1994)*; MIT Press: Cambridge, MA, USA, 1994; p 3.
49. Hinton, G.E.; Zemel, R.S. Minimizing Description Length in an Unsupervised Neural Network. Available online: https://www.cs.toronto.edu/~fritz/absps/mdlnn.pdf (accessed on 30 November 2019).
50. Deco, G.; Finnoff, W.; Zimmermann, H.G. Elimination of Overtraining by a Mutual Information Network. In *ICANN'93*; Gielen, S., Kappen, B., Eds.; Springer: London, UK, 1993; pp. 744–749. [CrossRef]
51. Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.A. Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 1096–1103.
52. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. In Proceedings of the International Conference on Learning Representations (ICLR 2014), Banff, AB, Canada, 14–16 April 2014.
53. Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; Lerchner, A. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In Proceedings of the International Conference on Learning Representations (ICLR 2017), Toulon, France, 24–26 April 2017.
54. Alemi, A.; Poole, B.; Fischer, I.; Dillon, J.; Saurous, R.A.; Murphy, K. Fixing a Broken ELBO. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 159–168.
55. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015.
56. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.
57. Pace, R.K.; Barry, R. Sparse spatial autoregressions. *Stat. Probab. Lett.* **1997**, *33*, 291–297. [CrossRef]
58. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).