Journal of Machine Learning Research 26 (2025) 1-49 Submitted 2/24; Revised 4/25; Published 8/25

Deep Variational Multivariate Information Bottleneck - A Framework for Variational Losses

Eslam Abdelaleem [*] [†] ESLAM.ABDELALEEM@EMORY.EDU
Department of Physics
Emory University
Atlanta, GA 30322, USA

Ilya Nemenman ILYA.NEMENMAN@EMORY.EDU
Departments of Physics and Biology
Initiative in Theory and Modeling of Living Systems
Emory University
Atlanta, GA 30322, USA

K. Michael Martini [*] [‡] KARL.MICHAEL.MARTINI@EMORY.EDU
Department of Physics
Emory University
Atlanta, GA 30322, USA

Editor: Quanquan Gu

Abstract

Variational dimensionality reduction methods are widely used for their accuracy, generative capabilities, and robustness. We introduce a unifying framework that generalizes both such as traditional and state-of-the-art methods. The framework is based on an interpretation of the multivariate information bottleneck, trading off the information preserved in an encoder graph (defining what to compress) against that in a decoder graph (defining a generative model for data). Using this approach, we rederive existing methods, including the deep variational information bottleneck, variational autoencoders, and deep multiview information bottleneck. We naturally extend the deep variational CCA (DVCCA) family to beta-DVCCA and introduce a new method, the deep variational symmetric information bottleneck (DVSIB). DSIB, the deterministic limit of DVSIB, connects to modern contrastive learning approaches such as Barlow Twins, among others. We evaluate these methods on Noisy MNIST and Noisy CIFAR-100, showing that algorithms better matched to the structure of the problem like DVSIB and beta-DVCCA produce better latent spaces as measured by classification accuracy, dimensionality of the latent variables, sample efficiency, and consistently outperform other approaches under comparable conditions. Additionally, we benchmark against state-of-the-art models, achieving superior or competitive accuracy. Our results demonstrate that this framework can seamlessly incorporate diverse multi-view representation learning algorithms, providing a foundation for designing novel, problem-specific loss functions.

Keywords: Information Bottleneck, Symmetric Information Bottleneck, Variational Methods, Generative Models, Dimensionality Reduction, Data Efficiency

[*]. Equal contribution
[†]. Currently eabdelaleem3@gatech.edu at the Schools of Physics and Psychology - Georgia Institute of Technology
[‡]. Corresponding author

1. Introduction

Large dimensional multi-modal datasets are abundant in multimedia systems utilized for language modeling (Xu et al., 2016; Zhou et al., 2018; Rohrbach et al., 2017; Sanabria et al., 2018; Goyal et al., 2017; Wang et al., 2018; Hendrycks et al., 2020), neural control of behavior studies (Steinmetz et al., 2021; Urai et al., 2022; Krakauer et al., 2017; Pang et al., 2016), multi-omics approaches in systems biology (Clark et al., 2013; Zheng et al., 2017; Svensson et al., 2018; Huntley et al., 2015; Lorenzi et al., 2018), and many other domains. Such data come with the curse of dimensionality, making it hard to learn the relevant statistical correlations from samples. The problem is made even harder by the data often containing information that is irrelevant to the specific questions one asks. To tackle these challenges, a myriad of supervised and unsupervised dimensionality reduction (DR) methods have emerged. By preserving certain aspects of the data while discarding the remainder, DR can decrease the complexity of the problem, yield clearer insights, and provide a foundation for more refined modeling approaches.

DR techniques span linear methods like Principal Component Analysis (PCA) (Hotelling, 1933), Partial Least Squares (PLS) (Wold et al., 2001), Canonical Correlations Analysis (CCA) (Hotelling, 1936), and regularized CCA (Vinod, 1976; Årup Nielsen et al., 1998), as well as nonlinear approaches, including Autoencoders (AE) (Hinton and Salakhutdinov, 2006), Deep CCA (Andrew et al., 2013), Deep Canonical Correlated AE (Wang et al., 2015), Correlational Neural Networks (Chandar et al., 2016), Deep Generalized CCA (Benton et al., 2017), and Deep Tensor CCA (Wong et al., 2021). Of particular interest to us are variational methods, such as Variational Autoencoders (VAE) (Kingma and Welling, 2014), beta-VAE (Higgins et al., 2016), Joint Multimodal VAE (JMVAE) (Suzuki et al., 2016), Deep Variational CCA (DVCCA) (Wang et al., 2016), Deep Variational Information Bottleneck (DVIB) (Alemi et al., 2017), Variational Mixture-of-experts AE (Shi et al., 2019), and Multiview Information Bottleneck (Federici et al., 2020b). These DR methods use deep neural networks and variational approximations to learn robust and accurate representations of the data, while, at the same time, often serving as generative models for creating samples from the learned distributions.

There are many theoretical derivations and justifications for variational DR methods (Kingma and Welling, 2014; Higgins et al., 2016; Suzuki et al., 2016; Wang et al., 2016; Karami and Schuurmans, 2021; Qiu et al., 2022; Alemi et al., 2017; Bao, 2021; Lee and Van der Schaar, 2021; Wang et al., 2019; Wan et al., 2021; Federici et al., 2020a; Huang et al., 2022; Hu et al., 2020). This diversity of derivations, while enabling adaptability, often leaves researchers with no principled ways for choosing a method for a particular application, for designing new methods with distinct assumptions, or for comparing methods to each other.

Here, we introduce the Deep Variational Multivariate Information Bottleneck (DVMIB) framework, offering a unified mathematical foundation for many variational—and deterministic—DR methods. Our framework is grounded in the multivariate information bottleneck loss function (Tishby et al., 2000; Friedman et al., 2013). This loss, amenable to approximation through upper and lower variational bounds, provides a system for implementing diverse DR variants using deep neural networks. We demonstrate the framework's efficacy by deriving the loss functions of many existing DR methods starting from the same principles. These include well-known methods, such as AEs (Hinton and Salakhutdinov, 2006), VAEs (Kingma and Welling, 2014), and state-of-the-art methods such as Contrastive Language-Image Pretraining (CLIP) (Radford et al., 2021) and Barlow Twins (Zbontar et al., 2021). Furthermore, our framework naturally allows the adjustment of trade-off parameters, leading to generalizations of these existing methods. For instance, we generalize DVCCA (Wang et al., 2016) to β-DVCCA. The framework further allows us to introduce and implement in software novel DR methods. We view the DVMIB framework, with its uniform information bottleneck language, conceptual clarity of translating statistical dependencies in data via graphical models of encoder and decoder structures into variational losses, the ability to unify existing approaches, and easy adaptability to new scenarios as one of the main contributions of our work.

Beyond its unifying role, our framework offers a principled approach for deriving problem-specific loss functions using domain-specific knowledge. Thus, we anticipate its application for multi-view representation learning across diverse fields. To illustrate this, we use the framework to derive a novel dimensionality reduction method, the Deep Variational Symmetric Information Bottleneck (DVSIB), which compresses two random variables into two distinct latent variables that are maximally informative about one another. This new method produces better representations of classic datasets than previous approaches. The introduction of DVSIB is another major contribution of our paper.

In summary, our paper makes the following contributions to the field:

1. **Introduction of the Variational Multivariate Information Bottleneck Framework:** We provide both intuitive and mathematical insights into this framework, establishing a robust foundation for further exploration.

2. **Rederivation and Generalization of Existing Methods within a Common Framework:** We demonstrate the versatility of our framework by systematically rederiving and generalizing various existing methods from the literature, showcasing the framework's ability to unify diverse approaches.

3. **Design of a Novel Method — Deep Variational Symmetric Information Bottleneck (DVSIB):** Employing our framework, we introduce DVSIB as a new method, contributing to the growing repertoire of techniques in variational dimensionality reduction. The method constructs high-accuracy latent spaces from substantially fewer samples than comparable approaches. Additionally, its deterministic version, the Deterministic Symmetric Information Bottleneck (DSIB), can be mapped to a plethora of state-of-the-art methods including CLIP and Barlow Twins.

The paper is structured as follows: First, in Sec. 2, we introduce the underlying mathematics and an implementation of the DVMIB framework. Then, in Sec. 3, we explain how to use the framework to generate new DR methods. In this section, in Tbl. 1, we present several known and newly variational DR methods, illustrating how easily they can be derived within the framework. In the Results (Sec. 4), as a proof of concept, we first benchmark *simple* computational implementations of the methods in Tbl. 1 against the Noisy MNIST dataset (Sec. 4.1). We then examine whether the trends observed in Noisy MNIST hold for more complex datasets, i.e., Noisy CIFAR-100, and more complex neural network architectures, namely CNNs (Sec. 4.2). Next, we demonstrate how the framework extends to more advanced, state-of-the-art methods (Sec. 4.3). The Appendices provide a detailed treatment of all terms in the different loss functions introduced (Appx. A). Appendix B.2 discusses auxiliary private variable models. Additional details, including visualizations, and a comprehensive performance analysis of many methods on Noisy MNIST are available in Appx. C. Further results for Noisy CIFAR-100 are provided in Appx. D.

2. Multivariate Information Bottleneck Framework

We represent DR problems similar to the Multivariate Information Bottleneck (MIB) of Friedman et al. (2013), which is a generalization of the more traditional Information Bottleneck algorithm (Tishby et al., 2000) to multiple variables. The reduced representation is achieved as a trade-off between two Bayesian networks. Bayesian networks are directed acyclic graphs that provide a factorization of the joint probability distribution, $P(X_1, X_2, X_3, .., X_N) = \prod_{i=1}^{N} P(X_i|Pa_{X_i}^G)$, where $Pa_{X_i}^G$ is the set of parents of X_i in graph G. The multiinformation (Studený and Vejnarová, 1998) of a Bayesian network is defined as the Kullback-Leibler divergence between the joint probability distribution and the product of the marginals, and it serves as a measure of the total correlations among the variables, $I(X_1, X_2, X_3, ..., X_N) = D_{KL}(P(X_1, X_2, X_3, ..., X_N)\|P(X_1)P(X_2)P(X_3)...P(X_N))$. For a Bayesian network, the multiinformation reduces to the sum of all the local informations $I(X_1, X_2, ..X_N) = \sum_{i=1}^{N} I(X_i; Pa_{X_i}^G)$ (Friedman et al., 2013).

The first of the Bayesian networks is an encoder (compression) graph, which models how compressed (reduced, latent) variables are obtained from the observations. The second network is a decoder graph, which specifies a generative model for the data from the compressed variables, i.e., it is an alternate factorization of the distribution. In MIB, the information of the encoder graph is minimized, ensuring strong compression (corresponding to the approximate posterior). The information of the decoder graph is maximized, promoting the most accurate model of the data (corresponding to maximizing the log-likelihood). As in IB (Tishby et al., 2000), the trade-off between the compression and reconstruction is controlled by a trade-off parameter β:

$$L = I_{\text{encoder}} - \beta I_{\text{decoder}}. \tag{1}$$

In this work, our key contribution is in writing an explicit variational loss for typical information terms found in both the encoder and the decoder graphs. All terms in the decoder graph use samples of the compressed variables as determined from the encoder graph. If there are two terms that correspond to the same information in Eq. (1), one from each of the graphs, they do not cancel each other since they correspond to two different variational expressions. For pedagogical clarity, we do this by first analyzing the Symmetric Information Bottleneck (SIB), a *special case* of MIB. We derive the bounds for three types of information terms in SIB, which we then use as building blocks for all other variational MIB methods in subsequent sections.

2.1 Deep Variational Symmetric Information Bottleneck

The Deep Variational Symmetric Information Bottleneck (DVSIB) simultaneously reduces a pair of datasets X and Y into two separate lower dimensional compressed versions Z_X and Z_Y. These compressions are done at the same time to ensure that the latent spaces are maximally informative about each other. The joint compression is known to decrease data set size requirements compared to individual ones (Martini and Nemenman, 2023). Having distinct latent spaces for each modality usually helps with interpretability. For example, X could be the neural activity of thousands of neurons, and Y could be the recordings of joint angles of the animal. Rather than one latent space representing both, separate latent spaces for the neural activity and the joint angles are sought. By maximizing compression as well as $I(Z_X, Z_Y)$, one constructs the latent spaces that capture only the neural activity pertinent to joint movement and only the movement that is correlated with the neural activity (cf. Pang et al. (2016)). Many other applications could benefit from a similar DR approach.

In Fig. 1, we define two Bayesian networks for DVSIB, G_{encoder} and G_{decoder}. G_{encoder} encodes the compression of X to Z_X and Y to Z_Y. It corresponds to the factorization $p(x, y, z_x, z_y) = p(x, y)p(z_x|x)p(z_y|y)$ and the resultant $I_{\text{encoder}} = I^E(X;Y) + I^E(X;Z_X) + I^E(Y;Z_Y)$. The $I^E(X, Y)$ term does not depend on the compressed variables, does not affect the optimization problem, and hence is discarded in what follows. G_{decoder} represents a generative model for X and Y given the compressed latent variables Z_X and Z_Y. It corresponds to the factorization $p(x, y, z_x, z_y) = p(z_x)p(z_y|z_x)p(x|z_x)p(y|z_y)$ and the resultant $I_{\text{decoder}} = I^D(Z_X;Z_Y) + I^D(X;Z_X) + I^D(Y;Z_Y)$. Combing the informations from both graphs and using Eq. (1), we find the SIB loss:

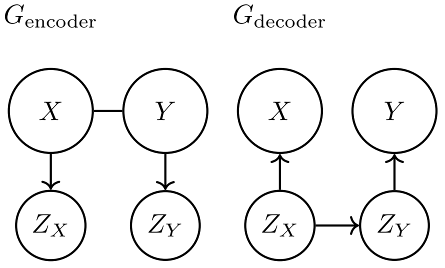


Figure 1: The encoder and decoder graphs for DVSIB.

$$L_{\text{SIB}} = I^E(X;Z_X) + I^E(Y;Z_Y) - \beta \left(I^D(Z_X;Z_Y) + I^D(X;Z_X) + I^D(Y;Z_Y) \right). \tag{2}$$

Note that information in the encoder terms is minimized, and information in the decoder terms is maximized. Thus, while it is tempting to simplify Eq. (2) by canceling $I^E(X;Z_X)$ and $I^D(X;Z_X)$, this would be a mistake. Indeed, these terms come from different factorizations: the encoder corresponds to learning $p(z_x|x)$, and the decoder to $p(x|z_x)$.

We now follow a procedure and notation similar to Alemi et al. (2017) and construct variational bounds on all I^E and I^D terms. Terms without leaf nodes, i. e., $I^D(Z_X, Z_Y)$, require new approaches.

2.2 Variational bounds on DVSIB encoder terms

The information $I^E(Z_X;X)$ corresponds to compressing the random variable X to Z_X. Since this is an encoder term, it needs to be minimized in Eq. (2). Thus, we seek a variational bound $I^E(Z_X;X) \leq \tilde{I}^E(Z_X;X)$, where $\tilde{I}^E$ is the variational version of I^E, which can be implemented using a deep neural network. We find $\tilde{I}^E$ by using the positivity of the Kullback–Leibler divergence. We make $r(z_x)$ be a variational approximation to $p(z_x)$. Then $D_{\mathrm{KL}}(p(z_x)\|r(z_x)) \geq 0$, so that $-\int dz_x p(z_x)\ln(p(z_x)) \leq -\int dz_x p(z_x)\ln(r(z_x))$. Thus, $-\int dx dz_x p(z_x,x)\ln(p(z_x)) \leq -\int dx dz_x p(z_x,x)\ln(r(z_x))$. We then add $\int dx dz_x p(z_x,x)\ln(p(z_x|x))$ to both sides and find:

$$I^E(Z_X;X) = \int dx dz_x p(z_x,x)\ln\left(\frac{p(z_x|x)}{p(z_x)}\right) \leq \int dx dz_x p(z_x,x)\ln\left(\frac{p(z_x|x)}{r(z_x)}\right) \equiv \tilde{I}^E(Z_X;X). \tag{3}$$

We further simplify the variational loss by approximating $p(x) \approx \frac{1}{N}\sum_{i=1}^{N}\delta(x-x_i)$, so that:

$$\tilde{I}^E(Z_X;X) \approx \frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln\left(\frac{p(z_x|x_i)}{r(z_x)}\right) = \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z_x|x_i)\|r(z_x)). \tag{4}$$

The term $I^E(Y;Z_Y)$ can be treated in an analogous manner, resulting in:

$$\tilde{I}^E(Z_Y;Y) \approx \frac{1}{N}\sum_{i=1}^{N} D_{KL}(p(z_y|y_i)\|r(z_y)). \tag{5}$$

Note that, in the deterministic limit, $p(z_x|x) \to \delta(z_x - f(x))$. In this limit, $I^E(Z_X;X) \to H(Z_X)$, and the latter is nominally infinite, but diverges only logarithmically with the discretization/regularization scale in z_x, and linearly with the dimensionality of Z_X. Provided $f(x)$ is sufficiently smooth, it does not affect the (regularized) $H(Z_X)$. Hence, compression is enforced not by details of the embedding $z_x = f(x)$, but only by the dimensionality of the latent variable. This deterministic limit is relevant for AEs (Hinton and Salakhutdinov, 2006) and several other deterministic methods that we will discuss later (cf. Sec. 4.3).

2.3 Variational bounds on DVSIB decoder terms

The term $I^D(X;Z)$ corresponds to a decoder of X from the compressed variable Z_X. It is maximized in Eq. (2). Thus, we seek its variational version $\tilde{I}^D$, such that $I^D \geq \tilde{I}^D$. Here, $q(x|z_x)$ will serve as a variational approximation to $p(x|z_x)$. We use the positivity of the Kullback-Leibler divergence, $D_{\mathrm{KL}}(p(x|z_x)\|q(x|z_x)) \geq 0$, to find $\int dx\, p(x|z_x)\ln(p(x|z_x)) \geq \int dx\, p(x|z_x)\ln(q(x|z_x))$. This gives $\int dz_x dx\, p(x,z_x)\ln(p(x|z_x)) \geq \int dz_x dx\, p(x,z_x)\ln(q(x|z_x))$. We add the entropy of X to both sides to arrive at the variational bound:

$$I^D(X;Z_X) = \int dz_x dx\, p(x,z_x)\ln\frac{p(x|z_x)}{p(x)} \geq \int dz_x dx p(x,z_x)\ln\frac{q(x|z_x)}{p(x)} \equiv \tilde{I}^D(X;Z_X). \tag{6}$$

We further simplify $\tilde{I}^D$ by replacing $p(x)$ by samples, $p(x) \approx \frac{1}{N}\sum_{i}^{N}\delta(x-x_i)$ and using the $p(z_x|x)$ that we learned previously from the encoder:

$$\tilde{I}^D(X;Z_X) \approx H(X) + \frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln(q(x_i|z_x)). \tag{7}$$

Here $H(X)$ does not depend on $p(z_x|x)$ and, therefore, can be dropped from the loss. The variational version of $I^D(Y;Z_Y)$ is obtained analogously:

$$\tilde{I}^D(Y;Z_Y) \approx H(Y) + \frac{1}{N}\sum_{i=1}^{N}\int dz_y p(z_y|y_i)\ln(q(y_i|z_y)). \tag{8}$$

2.4 Variational Bounds on decoder terms not on a leaf - MINE

The variational bound above cannot be applied to the information terms that do not contain leaves in G_{decoder}. For SIB, this corresponds to the $I^D(Z_X, Z_Y)$ term. This information is maximized. To find a variational bound such that $I^D(Z_X, Z_Y) \geq \tilde{I}^D(Z_X, Z_Y)$, we use a simple approximation, the Mutal Information Neural Estimator (MINE) (Belghazi et al., 2018), which samples both Z_X and Z_Y from their respective variational encoders. Other mutual information estimators, such as I_{InfoNCE} (Oord et al., 2018), and I_{SMILE} (Song and Ermon, 2019) can be used as long as they are differentiable. Different estimators might be better suited for different problems, and we explore some effects of the choice later (cf. Sec. 4.2 & Sec. 4.3). However, for simple applications and for developing intuition, I_{MINE} is sufficient. We variationally approximate $p(z_x, z_y)$ as $p(z_x)p(z_y)e^{T(z_x,z_y)}/\mathcal{Z}_{\text{norm}}$, where $\mathcal{Z}_{\text{norm}} = \int dz_x dz_y p(z_x)p(z_y)e^{T(z_x,z_y)} = \mathbb{E}_{z_x\sim p(z_x), z_y\sim p(z_y)}[e^{T(z_x,z_y)}]$ is the normalization factor[1]. Here $T(z_x, z_y)$, a critic function, is parameterized by a neural network that takes in samples of the latent spaces z_x and z_y and returns a single number. We again use the positivity of the Kullback-Leibler divergence, $D_{\text{KL}}(p(z_x, z_y)\|p(z_x)p(z_y)e^{T(z_x,z_y)}/\mathcal{Z}_{\text{norm}}) \geq 0$, which implies $\int dz_x dz_y p(z_x, z_y)\ln(p(z_x, z_y)) \geq \int dz_x dz_y p(z_x, z_y)\ln\frac{p(z_x)p(z_y)e^{T(z_x,z_y)}}{\mathcal{Z}_{\text{norm}}}$. Subtracting $\int dz_x dz_y p(z_x, z_y)\ln(p(z_x)p(z_y))$ from both sides, we find:

$$I^D(Z_X;Z_Y) \geq \int dz_x dz_y p(z_x, z_y)\ln\frac{e^{T(z_x,z_y)}}{\mathcal{Z}_{\text{norm}}} \equiv \tilde{I}^D_{\text{MINE}}(Z_X;Z_Y). \tag{9}$$

2.5 Parameterizing the distributions and the reparameterization trick

$H(X)$, $H(Y)$, and $I(X,Y)$ do not depend on $p(z_x|x)$ and $p(z_y|y)$ and are dropped from the loss. Further, we can use any ansatz for the variational distributions we introduced. We choose parametric probability distribution families and learn the nearest distribution in these families consistent with the data. We assume $p(z_x|x)$ is a normal distribution with mean $\mu_{Z_X}(x)$ and a diagonal variance $\Sigma_{Z_X}(x)$. We learn the mean and the log variance as neural networks. We also assume that $q(x|z_x)$ is normal with a mean $\mu_X(z_x)$ and a unit variance.[2]. Finally, we assume that $r(z_x)$ is a standard normal distribution. We use the reparameterization trick to produce samples of $z_{x\,i,j} = z_{x_j}(x_i) = \mu(x_i) + \sqrt{\Sigma_{Z_X}(x_i)}\eta_j$ from $p(z_x|x_i)$, where η_j is drawn from a standard normal distribution (Kingma and Welling, 2014). We choose the same types of distributions for the corresponding z_y terms.

To sample from $p(z_x, z_y)$ we use $p(z_x, z_y) = \int dxdy\, p(z_x, z_y, x, y) = \int dxdy\, p(z_x|x)p(z_y|y) \times p(x,y) \approx \frac{1}{N}\sum_{i=1}^{N} p(z_x|x_i)p(z_y|y_i) = \frac{1}{NM^2}\sum_{i=1}^{N}(\sum_{j=1}^{M}\delta(z_x - z_{x\,i,j}))(\sum_{j=1}^{M}\delta(z_y - z_{y_{i,j}}))$, where $z_{x\,i,j} \in p(z_x|x_i)$ and $z_{y_{i,j}} \in p(z_y|y_i)$, and M is the number of new samples being generated. To sample from $p(z_x)p(z_y)$, we generate samples from $p(z_x, z_y)$ and scramble the generated entries z_x and z_y, destroying all correlations. With this, the components of the loss function become

[1] The term $\mathcal{Z}_{\text{norm}}$ can be challenging to implement when doing optimization over batches. For example, Belghazi et al. (2018) propose using an expected moving average over the batches. Further discussion can be found in Poole et al. (2019). Simple implementations for the MINE estimator ignore this issue, but newer implementations (cf. Song and Ermon (2019); Abdelaleem et al. (2025)) provide better estimates, as we discuss later in Appx. A.3.

[2] In principle, we could also learn the variance for this distribution, but practically we did not find the need for that. Also note that, because we assume a unit variance for the decoder, both probabilistic and deterministic decoders are functionally equivalent: one models an exact map $x(z_x)$, and the other uses the same neural network to represent the mean, $\mathbb{E}_x(x|z_x)$.

$$\tilde{I}^E(X;Z_X) \approx \frac{1}{2N}\sum_{i=1}^{N}\left[\mathrm{Tr}(\Sigma_{Z_X}(x_i)) + ||\vec{\mu}_{Z_X}(x_i)||^2 - k_{Z_X} - \ln\det(\Sigma_{Z_X}(x_i))\right], \tag{10}$$

$$\tilde{I}^D(X;Z_X) \approx \frac{1}{MN}\sum_{i,j=1}^{N,M} -\frac{1}{2}||(x_i - \mu_X(z_{x i,j}))||^2, \tag{11}$$

$$\tilde{I}^D_{\mathrm{MINE}}(Z_X;Z_Y) \approx \frac{1}{M^2N}\sum_{i,j_x,j_y=1}^{N,M,M}\Big[T(z_{x i,j_x}, z_{y i,j_y}) - \ln \mathcal{Z}_{\mathrm{norm}}\Big], \tag{12}$$

where $\mathcal{Z}_{\mathrm{norm}} = \mathbb{E}_{z_x\sim p(z_x), z_y\sim p(z_y)}[e^{T(z_x,z_y)}]$, k_{Z_X} is the dimension of Z_X, and the corresponding terms for Y are similar. Combining these terms results in the variational loss for DVSIB:

$$L_{\mathrm{DVSIB}} = \tilde{I}^E(X;Z_X) + \tilde{I}^E(Y;Z_Y) - \beta\left(\tilde{I}^D_{\mathrm{MINE}}(Z_X;Z_Y) + \tilde{I}^D(X;Z_X) + \tilde{I}^D(Y;Z_Y)\right). \tag{13}$$

3. Deriving other DR methods

The variational bounds used in DVSIB can be used to implement loss functions that correspond to other encoder-decoder graph pairs and hence to other DR algorithms.

3.1 DVSIB variants

One can build another variant of DVSIB, but without reconstruction: DVSIB-noRecon. This variant is obtained by modifying the decoder graph in Fig. 1, G_{decoder}, by removing the arrows that go from the embeddings Z_X, Z_Y to X, Y which removes the two decoder terms $\tilde{I}^D(X;Z_X), \tilde{I}^D(Y;Z_Y)$ from the loss function (Eq. 13). In addition, a generic decoder not-on-a-leaf can be used $\tilde{I}^D(Z_X;Z_Y)$.

$$L_{\text{DVSIB-noRecon}} = \tilde{I}^E(X;Z_X) + \tilde{I}^E(Y;Z_Y) - \beta\tilde{I}^D(Z_X;Z_Y). \tag{14}$$

The deterministic variant of DVSIB-noRecon, DSIB-noRecon, is obtained by using deterministic encoders instead of probabilistic encoders, $p(z_x|x) = \delta(\vec{z}_x - \vec{\mu}_{Z_X}(x))$ and $p(z_y|y) = \delta(\vec{z}_y - \vec{\mu}_{Z_Y}(y))$, in addition to taking the limit of $\beta \to \infty$.

$$L_{\text{DSIB-noRecon}} = -\tilde{I}^D(Z_X;Z_Y). \tag{15}$$

The deterministic variant will be useful in deriving other methods (cf. Sec 4.3).

3.2 Other common DR methods

Aside from DVSIB and its variants, the simplest other common DR method to derive using the Framework is the beta variational auto-encoder (Kingma and Welling, 2014). Here G_{encoder} consists of one term: X compressed into Z_X. Similarly G_{decoder} consists of one term: X decoded from Z_X (see Table 1). Using this simple set of Bayesian networks, we find the variational loss:

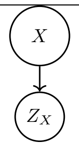
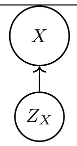
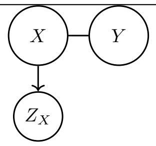
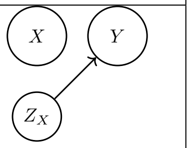
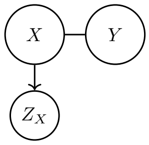
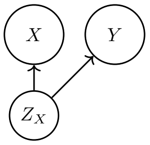
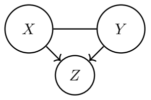
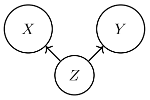
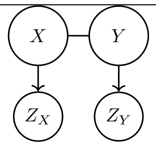
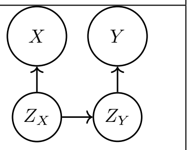
$$L_{\beta-\mathrm{VAE}} = \tilde{I}^E(X;Z_X) - \beta\tilde{I}^D(X;Z_X). \tag{16}$$

Both terms in Eq. (16) are the same as Eqs. (10, 11) and can be approximated and implemented by neural networks. Note that taking the deterministic limit of Eq. (16) results in a traditional AE.

If we have a supervising variable Y that we wish to reconstruct instead of X, modifying G_{decoder} accordingly results in DVIB, the Deep Variational Information Bottleneck (Alemi et al., 2017).

$$L_{\mathrm{DVIB}} = \tilde{I}^E(X;Z_X) - \beta\tilde{I}^D(Y;Z_X). \tag{17}$$

Table 1: Method descriptions, variational losses, and the Bayesian Network graphs for each DR method derived in our framework. See Appx.A for details.

Method Description	$G_{\textbf{encoder}}$	$G_{\textbf{decoder}}$
beta-VAE (Kingma and Welling, 2014; Higgins et al., 2016): Two independent Variational Autoencoder (VAE) models trained, one for each view, X and Y (only X graphs/loss shown). $L_{\text{VAE}} = \tilde{I}^E(X; Z_X) - \beta \tilde{I}^D(X; Z_X)$		
DVIB (Alemi et al., 2017): Two bottleneck models trained, one for each view, X and Y, using the other view as the supervising signal. (Only X graphs/loss shown). $L_{\text{DVIB}} = \tilde{I}^E(X; Z_X) - \beta \tilde{I}^D(Y; Z_X)$		
beta-DVCCA: Similar to DVIB (Alemi et al., 2017), but with reconstruction of both views. Two models trained, compressing either X or Y, while reconstructing both X and Y. (Only X graphs/loss shown). $L_{\text{DVCCA}} = \tilde{I}^E(X; Z_X) - \beta(\tilde{I}^D(Y; Z_X) + \tilde{I}^D(X; Z_X))$ **DVCCA** (Wang et al., 2016): β-DVCCA with $\beta = 1$.		
beta-joint-DVCCA: A single model trained using a concatenated variable $[X, Y]$, learning one latent representation Z. $L_{\text{jDVCCA}} = \tilde{I}^E((X, Y); Z) - \beta(\tilde{I}^D(Y; Z) + \tilde{I}^D(X; Z))$ **joint-DVCCA** (Wang et al., 2016): β-jDVCCA with $\beta = 1$.		
DVSIB: A symmetric model trained, producing Z_X and Z_Y. $L_{\text{DVSIB}} = \tilde{I}^E(X; Z_X) + \tilde{I}^E(Y; Z_Y)$ $-\beta \left(\tilde{I}^D_{\text{MINE}}(Z_X; Z_Y) + \tilde{I}^D(X; Z_X) + \tilde{I}^D(Y; Z_Y) \right)$		

Similarly, we can re-derive the DVCCA family of losses (Wang et al., 2016), Table 1. Here G_{encoder} is X compressed into Z_X. G_{decoder} reconstructs both X and Y from the same compressed latent space Z_X. In fact, our loss function is more general than the DVCCA loss and has an additional compression-reconstruction trade-off parameter β. We call this more general loss β-DVCCA, and the original DVCCA emerges when $\beta = 1$:

$$L_{\text{DVCCA}} = \tilde{I}^E(X; Z_X) - \beta(\tilde{I}^D(Y; Z_X) + \tilde{I}^D(X; Z_X)). \tag{18}$$

Using the same library of terms as we found for DVSIB, Eqs. (10, 11), we find:

$$\begin{aligned} L_{\text{DVCCA}} \approx & \frac{1}{N} \sum_{i=1}^{N} D_{\text{KL}}(p(z_x|x_i) \| r(z_x)) \\ & - \beta \left(\frac{1}{N} \sum_{i=1}^{N} \int dz_x p(z_x|x_i) \ln(q(y_i|z_x)) + \frac{1}{N} \sum_{i=1}^{N} \int dz_x p(z_x|x_i) \ln(q(x_i|z_x)) \right). \end{aligned} \tag{19}$$

This is similar to the loss function of the deep variational CCA (Wang et al., 2016), but now it has a trade-off parameter β. It trades off the compression into Z_X against the reconstruction of X and Y from the compressed variable Z_X. This apparent small change of adding β have a significant impact on the accuracy of DVCCA methods, as shown in the results (Sec. 4.1 & Sec. 4.2)

Table 1 shows how our framework reproduces and generalizes other traditional DR losses (see Appx. A). Our framework naturally extends beyond two variables and more state-of-the-art methods as well (see Tbl. 3). It also extends to models with private variables (see Appx. B.2).

4. Results

4.1 Noisy MNIST

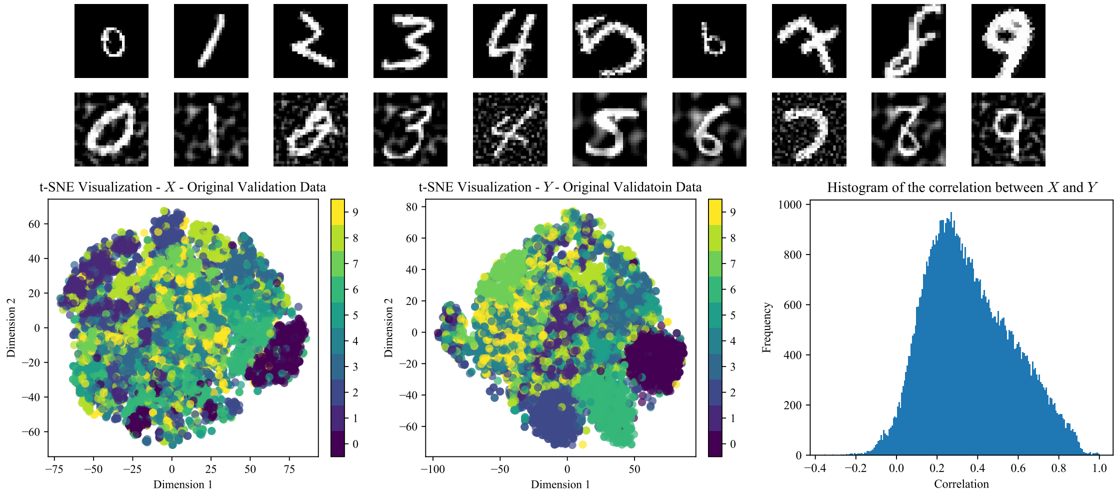


Figure 2: Dataset consisting of pairs of digits drawn from MNIST that share an identity. Top row, X: MNIST digits randomly scaled $(0.5 - 1.5)$ and rotated $(0 - \pi/2)$. Bottom row, Y: MNIST digits with a background Perlin noise. t-SNE of X and Y datasets (left and middle) shows poor separation by digit, and there is a wide range of correlation between X and Y (right).

To test our methods, we created a dataset inspired by the Noisy MNIST dataset (LeCun et al., 1998; Wang et al., 2015, 2016), consisting of two distinct views of data, both with dimensions of 28×28 pixels, cf. Fig. 2. The first view comprises the original image randomly rotated by an angle uniformly sampled between 0 and $\frac{\pi}{2}$ and scaled by a factor uniformly distributed between 0.5 and 1.5. The second view consists of the original image with an added background Perlin noise (Perlin, 1985) with the noise factor uniformly distributed between 0 and 1. Both image intensities are scaled to the range of $[0, 1)$. The dataset was shuffled within labels, retaining only the shared label identity between two images, while disregarding the view-specific details, i.e., the random rotation and scaling for X, and the correlated background noise for Y. The dataset, totaling $70,000$ images, was partitioned into training (80%), testing (10%), and validation (10%) subsets. Visualization via t-SNE (Hinton and Roweis, 2002) plots of the original dataset suggest poor separation by digit, and the two digit views have diverse correlations, making this a sufficiently hard problem.

The DR methods we evaluated include all methods from Tbl. 1. PCA and CCA (Hotelling, 1933, 1936) served as a baseline for linear dimensionality reduction. We emphasize that none of the algorithms were given labeled data. They had to infer compressed latent representations that presumably should cluster into ten different digits based simply on the fact that images come in pairs, and the (unknown) digit label is the only information that relates the two images.

Each method was trained for 100 epochs using fully connected neural networks with layer sizes $(\text{input\_dim}, 1024, 1024, (k_Z, k_Z))$, where k_Z is the latent dimension size, employing ReLU activations for the hidden layers. The input dimension (input_dim) was either the size of X (784) or the size of the concatenated $[X, Y]$ (1568). The last two layers of size k_Z represented the means and log(variance) learned. For the decoders, we employed regular decoders, fully connected neural networks with layer sizes $(k_Z, 1024, 1024, \text{output\_dim})$, using ReLU activations for

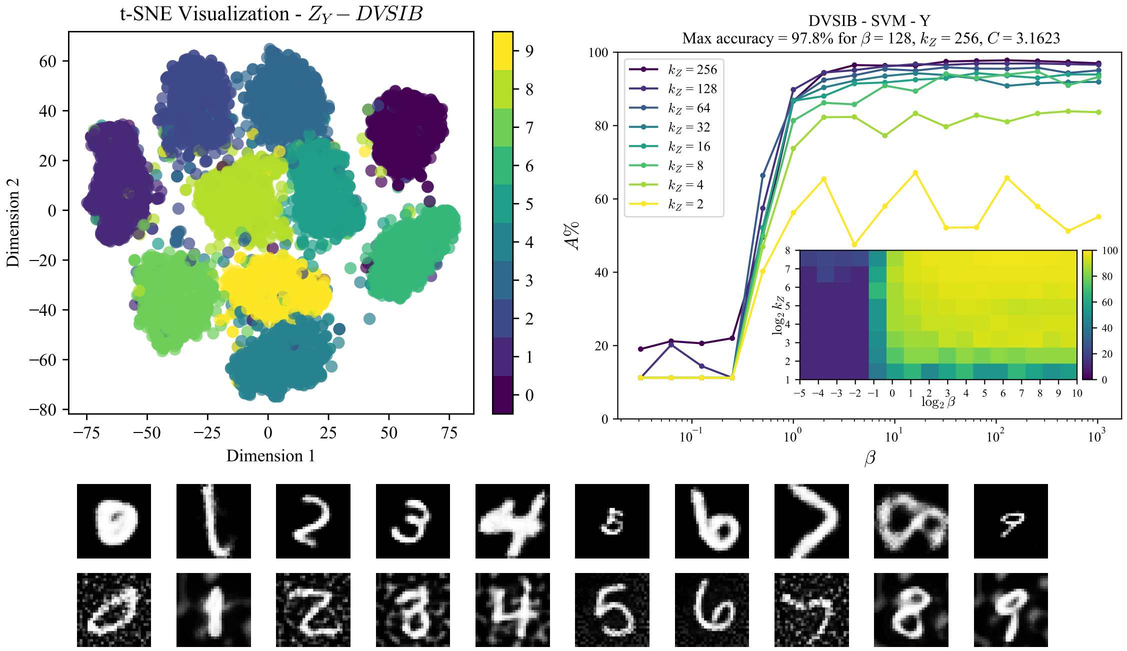


Figure 3: Top: t-SNE plot of the latent space Z_Y of DVSIB colored by the identity of digits. Top Right: Classification accuracy of an SVM trained on DVSIB's Z_Y latent space. The accuracy was evaluated for DVSIB with a parameter sweep of the trade-off parameter $\beta = 2^{-5}, ..., 2^{10}$ and the latent dimension $k_Z = 2^1, ..., 2^8$. The max accuracy was 97.8% for $\beta = 128$ and $k_Z = 256$. Bottom: Example digits generated by sampling from the DVSIB decoder, X and Y branches.

the hidden layers and sigmoid activation for the output layer. Again, the output dimension (output_dim) could either be the size of X (784) or the size of the concatenated $[X, Y]$ (1568).

The latent dimension (k_Z) could be k_{Z_X} or k_{Z_Y} for regular decoders, or $k_{Z_X} + k_{W_X}$ or $k_{Z_Y} + k_{W_Y}$ for decoders with private information. Additionally, another decoder based on the MINE estimator for estimating $I(Z_X, Z_Y)$, was used in DVSIB and DVSIB with private information. The critic of $I_{\text{MINE}}(Z_X, Z_Y)$ is a fully connected neural network with layer sizes $(k_{Z_X} + k_{Z_Y}, 1024, 1024, 1)$ and ReLU activations for the hidden layers. Optimization was conducted using the ADAM optimizer with default parameters.

Table 2: Maximum accuracy from a linear SVM and the optimal k_Z and β for variational DR methods reported on the Y (above the line) and the joint $[X, Y]$ (below the line) datasets. († fixed values)

Method	**Acc. %**	$k_{Z\,\mathbf{best}}$	95% k_Z	$\boldsymbol{\beta}_{\text{best}}$	95% β
Baseline	90.8	784†	-	-	-
PCA	90.5	256	[64,256*]	-	-
CCA	85.7	256	[32,256*]	-	-
β-VAE	96.3	256	[64,256*]	32	[2,1024*]
DVIB	90.4	256	[16,256*]	512	[8,1024*]
DVCCA	89.6	128	[16,256*]	1†	-
β-DVCCA	95.4	256	[64,256*]	16	[2,1024*]
DVSIB	**97.8**	256	[**8**,256*]	128	[2,1024*]
jBaseline	91.9	1568†	-	-	-
jDVCCA	92.5	256	[64,265*]	1†	-
β-jDVCCA	96.7	256	[16,265*]	256	[1,1024*]

To evaluate the methods, we trained them on the training portions of X and Y without exposure to the true labels. Subsequently, we used the trained encoders to compute Z_{train}, Z_{test}, and $Z_{\text{validation}}$ on the respective datasets. To assess the quality of the learned representations, we revealed the labels of Z_{train} and trained a linear SVM classifier with Z_{train} and $\text{labels}_{\text{train}}$. Hyper-parameter tuning of the classifier was performed to identify the optimal SVM slack parameter (C value), maximizing accuracy on $Z_{\text{validation}}$. This best classifier was then used to predict Z_{test}, yielding the reported accuracy. We also conducted classification experiments using fully connected neural networks, with detailed results available in Appx. C. For both SVM and the fully connected network, we find the baseline accuracy on the original training data and labels $(X_{\text{train}}, \text{labels}_{\text{train}})$ and $(Y_{\text{train}}, \text{labels}_{\text{train}})$, choose C based on the validation datasets, and report the results of the test datasets.

Using a linear SVM allows us to assess the linear separability of the clusters of Z_X and Z_Y obtained through DR methods. While neural networks can uncover complex nonlinear relationships that may lead to higher classification accuracy, a linear SVM ensures that classification performance reflects the quality of the embeddings themselves rather than the classifier's ability to compensate for any shortcomings in the DR methods. Additionally, if the embeddings are already linearly separable, they are likely to generalize more easily. Notably, we have also evaluated classification performance using neural networks, and the results (cf. Appx. C) show that DVSIB consistently achieves the highest or among the highest classification accuracies across both linear and nonlinear classifiers, further supporting the effectiveness of its embeddings.

Here, we focus on the results of the Y datasets (MNIST with correlated noise background); results for X are in Appx. C. A parameter sweep was performed to identify optimal k_Z values, ranging from 2^1 to 2^8 dimensions on $\log_2$ scale, as well as optimal β values, ranging from 2^{-5} to 2^{10}. For methods with private variables, k_{W_X} and k_{W_Y} were varied from 2^1 to 2^6 (results for these methods are available in Appx. B.2). The highest accuracy is reported in Tbl. 2, along with the optimal parameters used to obtain this accuracy.

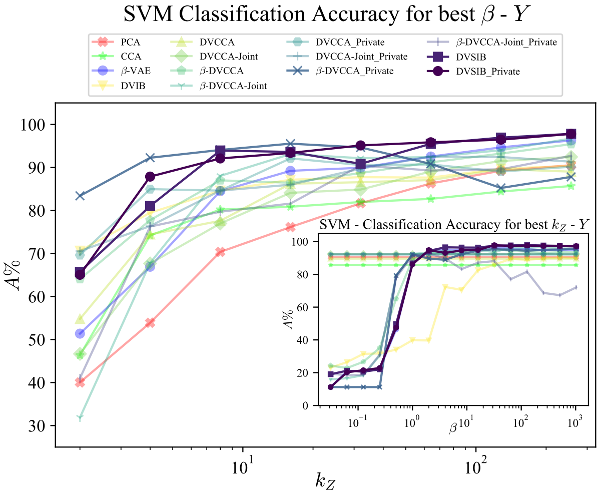


Figure 4: The best SVM classification accuracy curves for each method. Here, DVSIB and DVSIB-private achieved the highest accuracy, and, together with β-DVCCA-private, they performed best for low-dimensional latent spaces.

Additionally, for every method we find the range of β and the dimensionality k_Z of the latent variable Z_Y that gives 95% of the method's maximum accuracy. If the range includes the limits of the parameter, this is indicated by an asterisk.

Figure 3 shows a t-SNE plot of DVSIB's latent space, Z_Y, colored by the identity of digits. The resulting latent space has 10 clusters, each corresponding to one digit. The clusters are well separated and interpretable. Further, DVSIB's Z_Y latent space provides the best classification of digits using a linear method such as an SVM showing the latent space is linearly separable.

DVSIB maximum classification accuracy obtained for the linear SVM is 97.8%. Crucially, DVSIB maintains accuracy of at least 92.9% (95% of 97.8%) for $\beta \in [2, 1024^*]$ and $k_Z \in [8, 256^*]$. This accuracy is high compared to other methods and has a large range of parameters that maintain its ability to correctly capture information about the identity of the shared digit. Since DVSIB is a generative method, we also provided sample generated digits from the decoders that were trained from the model graph.

In Fig. 4, we show the highest SVM classification accuracy curves for each method. DVSIB and DVSIB-private tie for the best classification accuracy for Y. Together with β-DVCCA-private they have the highest accuracy for all dimensions of the latent space, k_Z. In theory, only one dimension should be needed to capture the identity of a digit, but our data sets also contain information about the rotation and scale for X and the strength of the background noise for Y. Y should then need at least two latent dimensions to be reconstructed and X should need at least three. Since DVSIB, DVSIB-private, and β-DVCCA-private performed with the best accuracy starting with the smallest k_Z, we conclude that methods with the encoder-decoder graphs that more closely match the structure of the data produce higher accuracy with lower dimensional latent spaces.

Next, in Fig. 5, we compare the sample training efficiency of DVSIB and β-VAE by training new instances of these methods on a geometrically increasing number of samples $n = [256, 339, 451, \ldots, \sim 42k, \sim 56k]$, consisting of 20 subsamples of the full training data (X, Y) to get $(X_{\text{train}_n}, Y_{\text{train}_n})$, where each larger subsample includes the previous one.

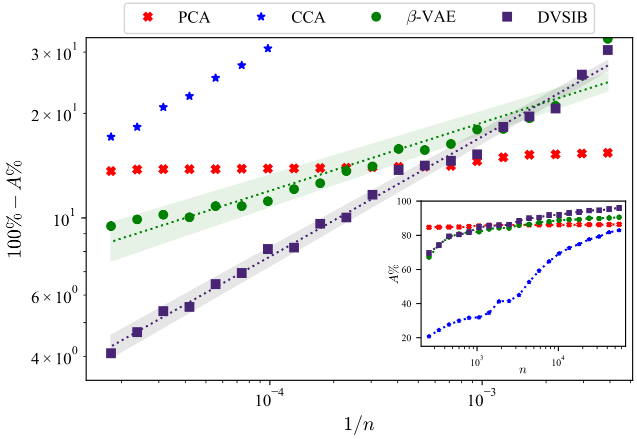


Figure 5: Classification accuracy (A) of DVSIB scales better with sample size (n). Main: a log-log plot of $100\% - A$ vs. $1/n$. The slope for fitted lines is 0.345 ± 0.007 for DVSIB and 0.196 ± 0.013 for β-VAE, indicating a faster increase in accuracy of DVSIB with n. Inset: same data plotted as A vs. n.

Each method was trained for 60 epochs, and we used $\beta = 1024$ (as defined by the DVMIB framework). Further, all reported results are with the latent space size $k_Z = 64$. We explored other numbers of training epochs and latent space dimensions (see Appx. C.5), but did not observe qualitative differences. We follow the same procedure as outlined earlier, using the 20 trained encoders for each method to compute Z_{train_n}, Z_{test}, and $Z_{\text{validation}}$ for the training, test, and validation datasets. As before, we then train and evaluate the classification accuracy of SVMs for the Z_Y representation learned by each method. Fig. 5, inset, shows the classification accuracy of each method as a function of the number of samples used in training. Again, CCA and PCA serve as linear methods baselines. PCA is able to capture the linear correlations in the dataset consistently, even at low sample sizes. However, it is unable to capture the nonlinearities of the data, and its accuracy does not improve with the sample size. Because of the iterative nature of the implementation of the PCA algorithm (Pedregosa et al., 2011), it is able to capture some linear correlations in a relatively low number of dimensions, which are sufficiently sampled even with small-sized datasets. Thus the accuracy of PCA barely depends on the training set size. CCA, on the other hand, does not work in the under-sampled regime (see Abdelaleem et al. (2024) for discussion of this). DVSIB performs uniformly better, at all training set sizes, than the β-VAE. Furthermore, DVSIB improves its quality faster, with a different sample size scaling. Specifically, DVSIB and β-VAE accuracy (A, measured in percent) appears to follow the scaling form $A = 100 - c/n^m$, where c is a constant, and the scaling exponent $m = 0.345 \pm 0.007$ for DVSIB, and 0.196 ± 0.013 for β-VAE. We illustrate this scaling in Fig. 5 by plotting a log-log plot of $100 - A$ vs $1/n$ and observing a linear relationship.

4.2 More complex tests: Noisy CIFAR-100 and CNN architecture

Next, we extend our analysis to more complex datasets and architectures to verify whether the trends observed in Noisy MNIST results hold. Similar to Noisy MNIST, we construct a Noisy CIFAR-100 dataset, which consists of $60,000$ RGB images of size 32×32, totaling 3072 pixels per image, with 100 classes and 600 images per class. We generate two distinct views in the same manner as Noisy MNIST: the first view (X) is randomly rotated by an angle uniformly sampled between 0 and $\frac{\pi}{2}$ and scaled by a factor uniformly drawn from $[0.5, 1.5]$. The second view (Y) incorporates background Perlin noise (Perlin, 1985), where the noise factor for each RGB channel is independently sampled from a uniform distribution over $[0, 1]$. Both views are rescaled to have pixel intensities in the range $[0, 1)$, and the dataset is shuffled within labels while preserving the shared label identity between the two transformed views (X and Y).

On the architecture side, instead of feedforward networks, we use a *simple* convolutional neural network (CNN) for the encoders and decoders of the different DR methods implemented. We observe classification accuracy trends similar to those in Noisy MNIST, where conv-DVSIB (and, to a great extent, conv-β-DVCCA) outperforms other methods with comparable architectures and computational costs. See Appx. D for details.

4.3 Towards the state of the art methods

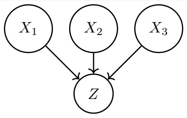
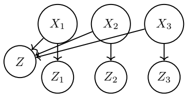
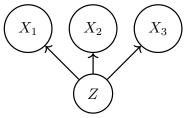
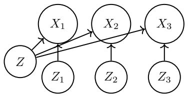
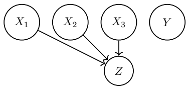
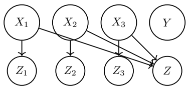
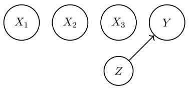
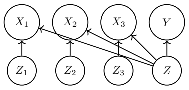
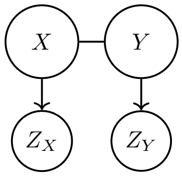
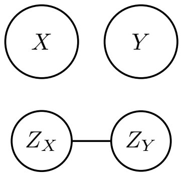
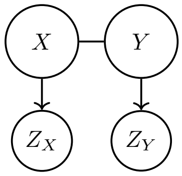
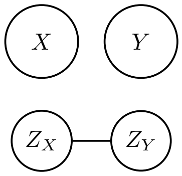
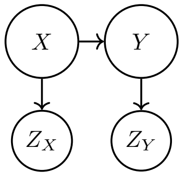
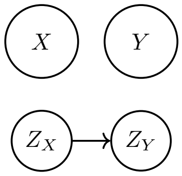
Having demonstrated that various common DR methods, Table 1, here we extend the approach to more recent, state-of-the-art methods. The fundamental trade-off between an encoder and a decoder graph can encapsulate widely used two-view learning approaches such as Barlow Twins Zbontar et al. (2021), CLIP Radford et al. (2021), and several other methods Bardes et al. (2024); Assran et al. (2023). Similarly, this framework can accommodate multi-view approaches, including Deep Multi-View Information Bottleneck methods Lee and Van der Schaar (2021); Wan et al. (2021); Huang et al. (2022), among others. We show that, by making simple assumptions (such as treating embeddings as deterministic and jointly Gaussian in the case of Barlow Twins), or by constructing the appropriate computational graphs, as in the Multi-View IB, these methods naturally fit within the DVMIB framework.

Table 3 provides an overview of these extended methods, with additional details in Appx. B.4. Moreover, we demonstrate that DVSIB, when adapted to match the structure and complexity of methods such as Barlow Twins and trained under comparable conditions, achieves similar or even superior accuracy. For details, see Appx. E.

5. Conclusion

We developed an MIB-based framework for deriving variational loss functions for DR applications. We demonstrated the use of this framework by developing a novel variational method, DVSIB. DVSIB compresses the variables X and Y into latent variables Z_X and Z_Y respectively, while maximizing the information between Z_X and Z_Y. The method generates two distinct latent spaces—a feature highly sought after in various applications—but it accomplishes this with superior data efficiency, compared to other methods. The example of DVSIB demonstrates the process of deriving variational bounds for terms present in all examined DR methods. A comprehensive library of typical terms is included in Appx. A for reference, which can be used to derive additional DR methods. Further, we (re)-derive several DR methods, as outlined in Tables 1&3. These include well-known techniques such as β-VAE, DVIB, DVCCA, and DVCCA-private, in addition to more complicated multiview techniques such as supervised and unsupervised MultiView IB, and state-of-the-art ones such as Barlow Twins and CLIP. MIB naturally introduces a trade-off parameter into the DVCCA family of methods, resulting in what we term the β-DVCCA DR methods, of which DVCCA is a special

Table 3: Method descriptions, variational losses, and the Bayesian Network graphs for more DR methods derived within the DVMIB framework.

Method Description	$G_{\textbf{encoder}}$	$G_{\textbf{decoder}}$
Unsupervised Multi-View IB (UMVIB): Methods like (Wan et al., 2021; Hu et al., 2020), where one compresses multiple views/modalities X_i into one (or more) latent variable Z (or Z_i) and reconstruct X_i from Z (or Z and Z_i). $L_{\text{UMVIB}_1} = \tilde{I}^E(X_1, X_2, X_3; Z) - \beta(\tilde{I}^D(X_1; Z) + \tilde{I}^D(X_2; Z) + \tilde{I}^D(X_3; Z))$ $L_{\text{UMVIB}_2} = \tilde{I}^E((X_1, X_2, X_3); Z) + \tilde{I}^E(X_1; Z_1) + \tilde{I}^E(X_2; Z_2) + \tilde{I}^E(X_3; Z_3) - \beta(\tilde{I}^D(X_1; (Z, Z_1)) + \tilde{I}^D(X_2; (Z, Z_2)) + \tilde{I}^D(X_3; (Z, Z_3)))$	 	 
Supervised Multi-View IB (SMVIB): Methods like (Lee and Van der Schaar, 2021; Wang et al., 2019; Huang et al., 2022; Grazioli et al., 2022), where one compresses multiple views/modalities X_i into a shared latent variable Z (or Z and Z_i), and uses Z to predict a supervisory signal/label Y (or in addition to using both Z and Z_i to reconstruct X_i as well). $L_{\text{SVMIB}_1} = \tilde{I}^E(X_1, X_2, X_3; Z) - \beta(\tilde{I}^D(Y; Z))$ $L_{\text{SMVIB}_2} = \tilde{I}^E((X_1, X_2, X_3); Z) + \tilde{I}^E(X_1; Z_1) + \tilde{I}^E(X_2; Z_2) + \tilde{I}^E(X_3; Z_3) - \beta(\tilde{I}^D(X_1; (Z, Z_1)) + \tilde{I}^D(X_2; (Z, Z_2)) + \tilde{I}^D(X_3; (Z, Z_3)) + \tilde{I}^D(Z; Y))$	 	 
Barlow Twins (BT)/DSIB (Zbontar et al., 2021): Two different views X and Y are compressed using the same deterministic compression function $\mu(\cdot)$ into Z_X and Z_Y respectively ($p(z_x\|x) = \delta(z_x - \mu(x))$ and $p(z_y\|y) = \delta(z_y - \mu(y))$). One then optimizes the cross-correlation matrix, C, between Z_X and Z_Y to be as close to identity as possible, which is equivalent to maximizing the information between $I(Z_X, Z_Y)$, when Z_X and Z_Y are jointly Gaussian. $L_{\text{DSIB-noRecon}} = -\tilde{I}^D(Z_X; Z_Y) = \frac{1}{2}\ln(\det(I - CC^T))$ Same minimum as: $L_{\text{BT}} = \sum_i (1 - C_{ii})^2 + \lambda \sum_{i,j} C_{ij}^2$		
CLIP/DSIB (Radford et al., 2021) : Two different views X and Y are compressed using different deterministic encoder functions into Z_X and Z_Y respectively, $p(z_x\|x) = \delta(\vec{z}_x - \vec{\mu}_{Z_X}(x))$ and $p(z_y\|y) = \delta(\vec{z}_y - \vec{\mu}_{Z_Y}(y))$. The CLIP loss is equivalent to $-I(Z_X, Z_Y)$ (and hence to the no reconstruction version of DSIB) if $p(z_x\|z_y) = p(z_y\|z_x)$ or $\frac{1}{2}(-I(Z_X, Z_Y) + H(Z_X, Z_Y))$ otherwise. $L_{\text{DSIB-noRecon}} = -\tilde{I}^D(Z_X; Z_Y)$. $L_{\text{CLIP}} = \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T)}{\frac{1}{N}\sum_{j=1}^{N}\exp(\vec{z}_{x_i}\cdot\vec{z}_{y_j}/T)}\right)$ $+\frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T)}{\frac{1}{N}\sum_{j=1}^{N}\exp(\vec{z}_{x_j}\cdot\vec{z}_{y_i}/T)}\right)$		
Possible Extensions: Methods like Bardes et al. (2024); Assran et al. (2023); Caron et al. (2021); Chen et al. (2020); Zhou et al. (2022); Baevski et al. (2022); Bardes et al. (2022); Zhang et al. (2022); Jia et al. (2021); Meng et al. (2022): probabilistically or deterministically compress X and Y into latent variables Z_X and Z_Y respectively and then learn a specific mapping between Z_Y and Z_X.		

case. We implement this new family of methods and show that it produces better latent spaces than DVCCA at $\beta = 1$, cf. Tbl. 2 and Tbl. 10.

We observe that methods that more closely match the structure of dependencies in the data can give better latent spaces as measured by the dimensionality of the latent space and the accuracy of reconstruction (see Figure 4). This makes DVSIB, DVSIB-private, and β-DVCCA-private perform the best. DVSIB and DVSIB-private both have separate latent spaces for X and Y. The private methods allow us to learn additional aspects about X and Y that are not important for the shared digit label, but allow reconstruction of the rotation and scale for X and the background noise of Y. We also found that DVSIB can make more efficient use of data when producing latent spaces as compared to β-VAEs and linear methods.

Our framework extends beyond variational approaches and provides a unified foundation for diverse learning methods, including dimensionality reduction, supervised and unsupervised multi-view learning, and self-supervised approaches. In particular, we have explicitly demonstrated how the deterministic limit of DVSIB recovers DSIB, and we have shown that this limit naturally maps onto state-of-the-art methods such as Barlow Twins and CLIP (Table. 3). Additionally, we have implemented convolutional and ResNet-based encoder-decoder architectures (Sec. 4.2, Sec. 4.3), illustrating that the framework is highly flexible: embedding functions can be easily swapped for more suitable architectures depending on the task. While we have not explicitly derived linear methods within our framework, prior work has shown that approaches like CCA can be viewed as special cases of the information bottleneck (Chechik et al., 2003), suggesting that they may also emerge naturally from our formulation. Overall, this versatility highlights the framework's ability to unify broad classes of learning techniques. With the provided tools and code, we aim to facilitate the adaptation of the framework to a wide range of problems.

Acknowledgments

EA and KMM contributed equally to this work. We thank Sean Ridout and Michael Pasek for providing feedback on the manuscript. EA and KMM also thank Ahmed Roman for useful and helpful discussions. EA is also grateful to Abdelrahman Nasser for his early help with coding when starting the project that eventually evolved into this work. IN is particularly grateful to the late Tali Tishby for many discussions about life, science, and information bottleneck over the years. This work was funded, in part, by NSF Grants Nos. 2010524 and 2014173, by the Simons Investigator award to IN, and the Simons-Emory International Consortium on Motor Control. We acknowledge support of our work through the use of the HyPER C3 cluster of Emory University's AI.Humanity Initiative.

Appendix

Table of Contents

Appendix A. Deriving and designing variational losses and their components

In the next two sections, we provide a library of typical terms found in encoder graphs, Appx. A.1, and decoder graphs, Appx. A.2. In Appx. B.1, we provide examples of combining these terms to produce variational losses corresponding to beta-VAE, DVIB, beta-DVCCA, beta-DVCCA-joint, beta-DVCCA-private, DVSIB, and DVSIB-private.

A.1 Encoder graph components

We expand Sec. 2.2 and present a range of common components found in encoder graphs across various DR methods, cf. Fig. (6).

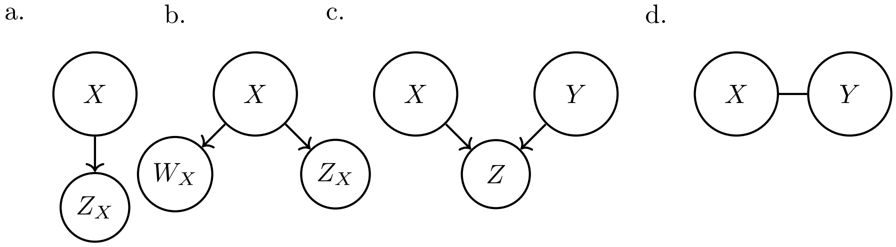


Figure 6: Encoder graph components.

a. This graph corresponds to compressing the random variable X to Z_X. Variational bounds for encoders of this type were derived in the main text in Sec. 2.2 and correspond to the loss:

$$\begin{aligned}\tilde{I}^E(X;Z_X) &= \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z_x|x_i)\|r(z_x)) \\ &\approx \frac{1}{2N}\sum_{i=1}^{N}\left[\mathrm{Tr}(\Sigma_{Z_X}(x_i)) + ||\vec{\mu}_{Z_X}(x_i)||^2 - k_{Z_X} - \ln\det(\Sigma_{Z_X}(x_i))\right]. \qquad (20)\end{aligned}$$

b. This type of encoder graph is similar to the first, but now with two outputs, Z_X and W_X. This corresponds to making two encoders, one for Z_X and one for W_X, $\tilde{I}^E(Z_X;X) + \tilde{I}^E(W_X;X)$, where

$$\tilde{I}^E(Z_X;X) \approx \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z_x|x_i)\|r(z_x)), \qquad (21)$$

$$\tilde{I}^E(W_X;X) \approx \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(w_x|x_i)\|r(w_x)). \qquad (22)$$

c. This type of encoder consists of compressing X and Y into a single variable Z. It corresponds to the information loss $I^E(Z;(X,Y))$. This again has a similar encoder structure to type (a), but X is replaced by a joint variable (X,Y). For this loss, we find a variational version:

$$\tilde{I}^E(Z;(X,Y)) \approx \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z|x_i,y_i)\|r(x_i,y_i)). \qquad (23)$$

d. This final type of an encoder term corresponds to information $I^E(X,Y)$, which is constant with respect to our minimization. In practice, we drop terms of this type.

A.2 Decoder graph components

In this section, we elaborate on the decoder graphs that happen in our considered DR methods, cf. Fig. (7).

All decoder graphs sample from their methods' corresponding encoder graph.

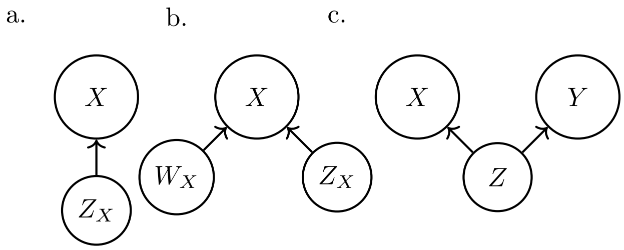


Figure 7: Decoder graph components.

a. In this decoder graph, we decode X from the compressed variable Z_X. Variational bounds for decoders of this type were derived in the main text, Sec. 2.3, and they correspond to the loss:

$$\begin{aligned}\tilde{I}^D(X;Z_X) &= H(X) + \frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln q(x_i|z_x)\\ &\approx H(X) + \frac{1}{MN}\sum_{i,j=1}^{N,M} -\frac{1}{2}||(x_i - \mu_X(z_{xi,j}))||^2,\end{aligned} \tag{24}$$

where $H(X)$ can be dropped from the loss since it doesn't change in optimization.

b. This type of decoder term is similar to that in part (a), but X is decoded from two variables simultaneously. The corresponding loss term is $I^D(X;(Z_X,W_X))$. We find a variational loss by replacing Z_X in part (a) by (Z_X,W_X):

$$\tilde{I}^D(X;(Z_X,W_X)) \approx H(X) + \frac{1}{N}\sum_{i=1}^{N}\int dz_x dw_x p(z_x,w_x|x_i)\ln(q(x_i|z_x,w_x)), \tag{25}$$

where, again, the entropy of X can be dropped.

c. This decoder term can be obtained by adding two decoders of type (a) together. In this case, the loss term is $I^D(X;Z) + I^D(Y;Z)$:

$$\begin{aligned}&\tilde{I}^D(X;Z) + \tilde{I}^D(Y;Z) \approx H(X) + H(Y)\\ &\qquad + \frac{1}{N}\sum_{i=1}^{N}\int dz p(z|x_i)\ln(q(x_i|z)) + \frac{1}{N}\sum_{i=1}^{N}\int dz p(z|y_i)\ln(q(y_i|z)),\end{aligned} \tag{26}$$

and the entropy terms can be dropped, again.

A.3 Internal decoders (decoders not on a leaf)

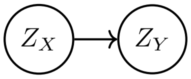


Figure 8: Internal Decoders

Decoders of this type were discussed in the main text in Sec. 2.4. They correspond to the information between latent variables Z_X and Z_Y. We use the MINE estimator to find variational bounds for such terms:

$$\tilde{I}^D_{\mathrm{MINE}}(Z_X; Z_Y) = \int dz_x dz_y p(z_x, z_y) \ln \frac{e^{T(z_x,z_y)}}{\mathcal{Z}_{\mathrm{norm}}} \approx \frac{1}{NM^2} \sum_{i,j_x,j_y=1}^{N,M,M} \Big[T(z_{x_{i,j_x}}, z_{y_{i,j_y}}) - \ln \mathcal{Z}_{\mathrm{norm}} \Big] . \tag{27}$$

We use $I_{\mathrm{MINE}}(X, Y)$ for the implementations in Sec. 4.1. It is also possible to estimate terms of this type using other mutual information estimators such as SMILE and InfoNCE. The SMILE estimator (Song and Ermon, 2019) is a clipped version of the MINE estimator. The SMILE estimator improves the robustness of MINE by clipping the joint to marginal density ratio between $e^{-\tau}$ and e^{τ}, where τ is some parameter that we set to 5 in our implementation:

$$I_{\mathrm{SMILE}}(X, Y) \geq \mathbb{E}_P[T(x, y)] - \log \Big[\mathbb{E}_Q \Big(\mathrm{clip}(e^{T(x,y)}, e^{-\tau}, e^{\tau}) \Big) \Big] . \tag{28}$$

Smaller τ decreases the variance, but at a cost of a larger bias. At $\tau \to \infty$, $I_{\mathrm{SMILE}} \to I_{\mathrm{MINE}}$. We use $I_{\mathrm{SMILE}}(X, Y)$ for the implementations mentioned in Sec. 4.2.

InfoNCE (Oord et al., 2018) is a mutual information estimator that uses a contrastive loss. When using a separable critic, it consists of two networks, g and h. These networks compute $g(Z_X)$ and $h(Z_Y)$, which are then combined via a dot product to form the critic function $f(Z_X, Z_Y) = g(X) \cdot h(Y)$. The InfoNCE estimate of mutual information is then given by

$$I_{\mathrm{InfoNCE}}(Z_X; Z_Y) \approx \mathbb{E} \left[\log \frac{e^{f(Z_X,Z_Y)}}{\sum_{Z'_Y} e^{f(Z_X,Z'_Y)}} \right] , \tag{29}$$

where the denominator sums over negative samples Z'_Y. This loss can easily be recognized as mutual information if we make the identification $e^{f(Z_X,Z_Y)} = p(Z_X|Z_Y)$. We use $I_{\mathrm{InfoNCE}}(X, Y)$ for the implementations mentioned in Sec. 4.3. A detailed treatment of the different estimators and best practices of using them can also be found in Abdelaleem et al. (2025).

Appendix B. Deriving and designing variational losses: detailed implementations

B.1 Two variable Losses

For completeness, we provide detailed implementations of methods outlined in Tbl. 1.

B.1.1 Beta Variational Auto-Encoder

A variational autoencoder (Kingma and Welling, 2014; Higgins et al., 2016) compresses X into a latent variable Z_X and then reconstructs X from the latent variable, cf. Fig. (10). The overall loss is a trade-off between the compression $I^E(X; Z_X)$ and the reconstruction $I^D(X; Z_X)$:

$$\begin{aligned} I^E(X; Z_X) - \beta I^D(X; Z_X) &\leq \tilde{I}^E(X; Z_X) - \beta \tilde{I}^D(X; Z_X) \\ &\lesssim \frac{1}{N} \sum_{i=1}^{N} D_{\mathrm{KL}}(p(z_x|x_i) \| r(z_x)) - \beta \left(H(X) + \frac{1}{N} \sum_{i=1}^{N} \int dz_x p(z_x|x_i) \ln(q(x_i|z_x)) \right) . \end{aligned} \tag{30}$$

$H(X)$ is a constant with respect to the minimization, and it can be omitted from the loss. Similar to the main text, DVSIB case, we make ansatzes for forms of each of the variational distributions. We choose parametric distribution families and learn the nearest distribution in these families consistent

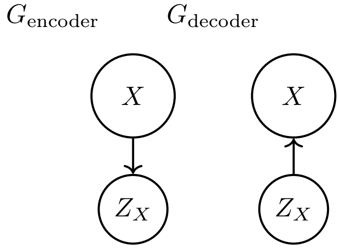


Figure 9: Encoder and decoder graphs for the beta-variational auto-encoder method

with the data. Specifically, we assume $p(z_x|x)$ is a normal distribution with mean $\mu_{Z_X}(X)$ and variance $\Sigma_{Z_X}(X)$. We learn the mean and the log-variance as neural networks. We also assume that $q(x|z_x)$ is normal with a mean $\mu_X(z_x)$ and a unit variance. Finally, we assume that $r(z_x)$ is drawn from a standard normal distribution. We then use the re-parameterization trick to produce samples of $z_{x_j}(x) = \mu(x) + \sqrt{\Sigma_{Z_X}(x)}\eta_j$ from $p(z_x|x)$, where η is drawn from a standard normal distribution. Overall, this gives:

$$L_{\text{VAE}} = \frac{1}{2N}\sum_{i=1}^{N}\left[\text{Tr}(\Sigma_{Z_X}(x_i)) + \vec{\mu}_{Z_X}(x_i)^T\vec{\mu}_{Z_X}(x_i) - k_{Z_X} - \ln\det(\Sigma_{Z_X}(x_i))\right]$$
$$- \beta\left(\frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M} -\frac{1}{2}(x_i - \mu_X(z_{x_j}))^T(x_i - \mu_X(z_{x_j}))\right). \quad (31)$$

This is the same loss as for a beta auto-encoder. However, following the convention in the Information Bottleneck literature (Tishby et al., 2000; Friedman et al., 2013), our β is the inverse of the one typically used for beta auto-encoders. A small β in our case results in a stronger compression, while a large β results in a better reconstruction.

B.1.2 Deep Variational Information Bottleneck

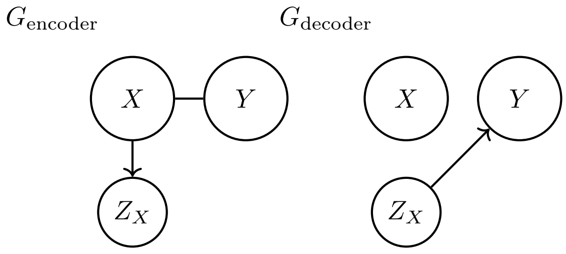


Figure 10: Encoder and decoder graphs for the Deep Variational Information Bottleneck.

Just as in the beta auto-encoder, we immediately write down the loss function for the information bottleneck. Here, the encoder graph compresses X into Z_X, while the decoder tries to maximize the information between the compressed variable and the relevant variable Y, cf. Fig. (10). The resulting loss function is:

$$L_{\text{IB}} = I^E(X;Y) + I^E(X;Z_X) - \beta I^D(Y;Z_X). \quad (32)$$

Here the information between X and Y does not depend on $p(z_x|x)$ and can dropped in the optimization.

Thus the Deep Variational Information Bottleneck (Alemi et al., 2017) becomes :

$$L_{\text{DVIB}} \quad \approx \quad \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z_x|x_i)\|r(z_x)) \quad - \quad \beta\left(\frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln(q(y_i|z_x))\right), \tag{33}$$

where we dropped $H(Y)$ since it doesn't change in the optimization.

As we have been doing before, we choose to parameterize all these distributions by Gaussians and their means and their log variances are learned by neural networks. Specifically, we parameterize $p(z_x|x) = N(\mu_{z_x}(x), \Sigma_{z_x})$, $r(z_x) = N(0, I)$, and $q(y|z_x) = N(\mu_Y, I)$. Again we can use the reparameterization trick and sample from $p(z_x|x_i)$ by $z_{x_j}(x) = \mu(x) + \sqrt{\Sigma_{z_x}(x)}\eta_j$ where η is drawn from a standard normal distribution.

B.1.3 Beta Deep Variational CCA

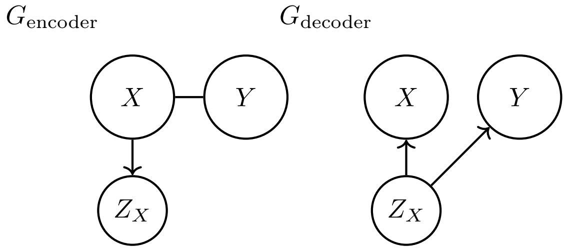


Figure 11: Encoder and decoder graphs for beta Deep Variational CCA.

beta-DVCCA, cf. Fig. 11, is similar to the traditional information bottleneck, but now X and Y are both used as relevance variables:

$$L_{\text{DVCCA}} = \tilde{I}^E(X;Y) + \tilde{I}^E(X;Z_X) - \beta(\tilde{I}^D(Y;Z_X) + \tilde{I}^D(X;Z_X)). \tag{34}$$

Using the same library of terms as before, we find:

$$\begin{aligned} L_{\text{DVCCA}} \approx & \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z_x|x_i)\|r(z_x)) \\ & - \beta\left(\frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln(q(y_i|z_x)) + \frac{1}{N}\sum_{i=1}^{N}\int dz_x p(z_x|x_i)\ln(q(x_i|z_x))\right). \end{aligned} \tag{35}$$

This is similar to the loss function of the deep variational CCA (Wang et al., 2016), but now it has a trade-off parameter β. It trades off the compression into Z against the reconstruction of X and Y from the compressed variable Z.

B.1.4 beta joint-Deep Variational CCA

Joint deep variational CCA (Wang et al., 2016), cf. Fig. 12, compresses (X, Y) into one Z and then reconstructs the individual terms X and Y,

$$L_{\text{DVCCA}} = I^E(X;Y) + I^E((X,Y);Z) - \beta(I^D(Y;Z) + I^D(X;Z)). \tag{36}$$

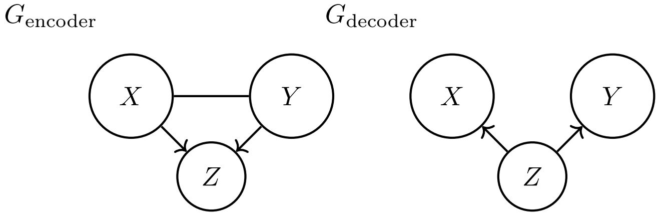


Figure 12: Encoder and decoder graphs for beta joint-Deep Variational CCA.

Using the terms we derived, the loss function is:

$$
\begin{aligned}
L_{\text{DVCCA}} \approx \frac{1}{N}\sum_{i=1}^{N} D_{KL}(p(z|x_i, y_i)\|r(z)) \\
- \beta \left(\frac{1}{N}\sum_{i=1}^{N} \int dz p(z|x_i) \ln(q(y_i|z)) + \frac{1}{N}\sum_{i=1}^{N} \int dz p(z|x_i) \ln(q(x_i|z)) \right). \quad (37)
\end{aligned}
$$

The information between X and Y does not change under the minimization and can be dropped.

B.2 Auxiliary private variable models

Most of the losses examined so far aim to capture the correlation between X and Y. If, however, one wishes additionally to separate out details about just X and just Y, it is helpful to introduce additional auxiliary private variables W_X, W_Y intended to capture these details. In DVCCA-private (beta-DVCCA-private) for example, the decoder factorization illustrates this clearly: $P(x|z, w_x)P(x|z, w_y)P(z)P(w_x)P(w_y)$. This shows that X is reconstructed from the private variable W_X that only has information about X and the shared variable Z, and similarly Y is reconstructed from the private variable W_Y which only has information about Y and the shared variable Z. In the Noisy MNIST dataset this type of factorization encourages Z to exclusively capture details about digit identity, W_X to capture details about scale and rotation, and W_Y details about noise. Similar results hold for beta-joint-DVCCA-private, and DVSIB-private. See C for Noisy MNIST visualizations.

B.2.1 beta (joint) Deep Variational CCA-private

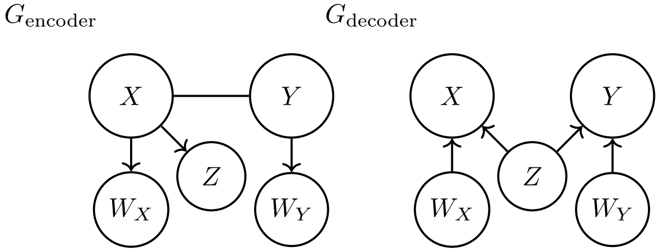
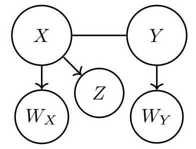
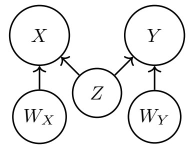
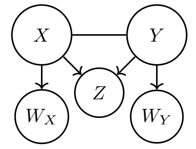
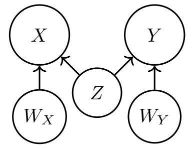
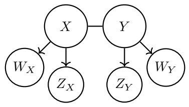
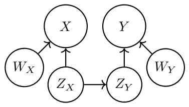


Figure 13: Encoder and decoder graphs for beta Deep Variational CCA-private

This is a generalization of the Deep Variational CCA Wang et al. (2016) to include private information, cf. Fig. 13. Here X is encoded into a shared latent variable Z and a private latent

Table 4: Method descriptions, variational losses, and the Bayesian Network graphs for each DR method with private variables derived in our framework. See Appx. A for details.

Method Description	$G_{\textbf{encoder}}$	$G_{\textbf{decoder}}$
beta-DVCCA-private: Two models trained, compressing either X or Y, while reconstructing both X and Y, and simultaneously learning private information W_X and W_Y. (Only X graphs/loss shown). $L_{\text{DVCCA-p}} = \tilde{I}^E(X;Z) + \tilde{I}^E(X;W_X) + \tilde{I}^E(Y;W_Y) - \beta(\tilde{I}^D(X;(W_X,Z)) + \tilde{I}^D(Y;(W_Y,Z)))$ **DVCCA-private** (Wang et al., 2016): β-DVCCA-p with $\beta = 1$.		
beta-joint-DVCCA-private: A single model trained using a concatenated variable $[X,Y]$, learning one latent representation Z, and simultaneously learning private information W_X and W_Y. $L_{\text{jDVCCA-p}} = \tilde{I}^E((X,Y);Z) + \tilde{I}^E(X;W_X) + \tilde{I}^E(Y;W_Y) - \beta(\tilde{I}^D(X;(W_X,Z)) + \tilde{I}^D(Y;(W_Y,Z)))$ **joint-DVCCA-private**(Wang et al., 2016): β-jDVCCA-p with $\beta = 1$.		
DVSIB-private: A symmetric model trained, producing Z_X and Z_Y, while simultaneously learning private information W_X and W_Y. $L_{\text{DVSIBp}} = \tilde{I}^E(X;W_X) + \tilde{I}^E(X;Z_X) +$ $\tilde{I}^E(Y;Z_Y) + \tilde{I}^E(Y;W_Y) -$ $\beta\left(\tilde{I}^D_{\text{MINE}}(Z_X;Z_Y) + \tilde{I}^D(X;(Z_X,W_X)) + \tilde{I}^D(Y;(Z_Y,W_Y))\right)$		

variable W_X. Similarly Y is encoded into the same shared variable and a different private latent variable W_Y. X is reconstructed from Z and W_X, and Y is reconstructed from Z and W_Y. In the joint version (X,Y) are compressed jointly in Z similar to the previous joint methods. What follows is the loss X version of beta Deep Variational CCA-private.

$$L_{\text{DVCCAp}} = I^E(X;Y) + I^E((X,Y);Z) + I^E(X;W_X) + I^E(Y;W_Y) - \beta(I^D(X;(W_X,Z)) + I^D(Y;(W_Y,Z))). \quad (38)$$

After the usual variational manipulations, this becomes:

$$\begin{aligned} L_{\text{DVCCAp}} \approx & \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z|x_i)\|r(z)) + \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(w_x|x_i)\|r(w_x)) \\ & + \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(w_y|y_i)\|r(w_y)) - \beta\left(\frac{1}{N}\sum_{i=1}^{N}\int dz dw_x p(w_x|x_i)p(z|x_i)\ln(q(y_i|z,w_x))\right. \\ & \left. + \frac{1}{N}\sum_{i=1}^{N}\int dz dw_y p(w_y|y_i)p(z|x_i)\ln(q(x_i|z,w_y))\right). \quad (39) \end{aligned}$$

B.2.2 Deep Variational Symmetric Information Bottleneck

This has been analyzed in detail in the main text, Sec. 2.1, and will not be repeated here.

B.2.3 Deep Variational Symmetric Information Bottleneck-private

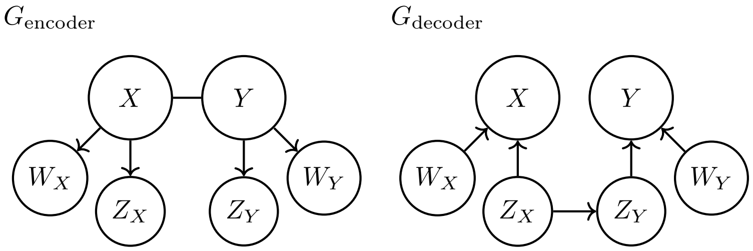


Figure 14: Encoder and decoder graphs for DVSIB-private.

This is a generalization of the Deep Variational Symmetric Information Bottleneck to include private information. Here X is encoded into a shared latent variable Z_X and a private latent variable W_X. Similarly, Y is encoded into its own shared Z_Y variable and a private latent variable W_Y. X is reconstructed from Z_X and W_X, and Y is reconstructed from Z_Y and W_Y. Z_X and Z_Y are constructed to be maximally informative about each another. This results in

$$\begin{aligned} L_{\text{DVSIBp}} = I^E(X;W_X) + I^E(X;Z_X) + I^E(Y;Z_Y) + I^E(Y;W_Y) \\ - \beta\left(I^D(Z_X;Z_Y) + I^D(X;(Z_X,W_X)) + I^D(Y;(Z_Y,W_Y))\right). \end{aligned} \tag{40}$$

After the usual variational manipulations, this becomes (see also main text):

$$\begin{aligned} L_{\text{DVSIBp}} \approx \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z_x|x_i)\|r(z_x)) + \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z_y|x_i)\|r(z_y)) \\ + \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(w_x|x_i)\|r(w_x)) + \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(w_y|y_i)\|r(w_y)) \\ - \beta\Bigg(\int dz_x dz_y p(z_x,z_y)\ln\frac{e^{T(z_x,z_y)}}{\mathcal{Z}_{\text{norm}}} + \frac{1}{N}\sum_{i=1}^{N}\int dz_y dw_y p(w_y|y_i)p(z_y|y_i)\ln(q(y_i|z_y,w_y)) \\ + \frac{1}{N}\sum_{i=1}^{N}\int dz_x dw_x p(w_x|x_i)p(z_x|x_i)\ln(q(x_i|z_x,w_x))\Bigg), \end{aligned} \tag{41}$$

where

$$\mathcal{Z}_{\text{norm}} = \int dz_x dz_y p(z_x)p(z_y)e^{T(z_x,z_y)}. \tag{42}$$

B.2.4 Private Variable Model Results on Noisy MNIST

Details of experiments can be found in Tbl. 5.

B.3 Multi-variable losses (more than two views/variables)

Several multi-variable losses that have appeared in the literature can be rederived within our framework. These methods can be broadly categorized into two groups: supervised multiview

Table 5: Maximum accuracy from a linear SVM and the optimal k_Z and β for variational DR methods with private variables reported on the Y (above the line) and the joint $[X, Y]$ (below the line) datasets. († fixed values)

Method	**Acc. %**	$k_{Z\textbf{best}}$	95% $k_{Z\textbf{range}}$	$\boldsymbol{\beta}_{\text{best}}$	95% $\beta_{\textbf{range}}$
Baseline	90.8	$784^\dagger$	-	-	-
DVCCA-p	92.1	16	[16,256*]	$1^\dagger$	-
β-DVCCA-p	95.5	16	[**4**,256*]	1024	[1,1024*]
DVSIB-p	**97.8**	256	[**8**,256*]	32	[2,1024*]
jBaseline	91.9	$1568^\dagger$	-	-	-
jDVCCA-p	92.5	64	[32,265*]	$1^\dagger$	-
β-jDVCCA-p	92.7	256	[**4**,265*]	2	[1,1024*]

information bottleneck (SMVIB) and unsupervised multiview information bottleneck (UMVIB). The key distinction between them is the presence of a supervising signal Y in addition to the multiple views or modalities of the data X_i. Here, we demonstrate how our DVMIB framework provides a natural way to extend and unify these methods.

B.3.1 Unsupervised Multi-View IB

Several approaches in the literature, such as Wan et al. (2021); Hu et al. (2020), focus on compressing multiple views or modalities X_i into one or more latent variables Z (or Z_i) and reconstructing X_i from Z (or both Z and Z_i). Within this framework, we can consider two primary formulations:

- **Single shared latent representation (UMVIB$_1$):**

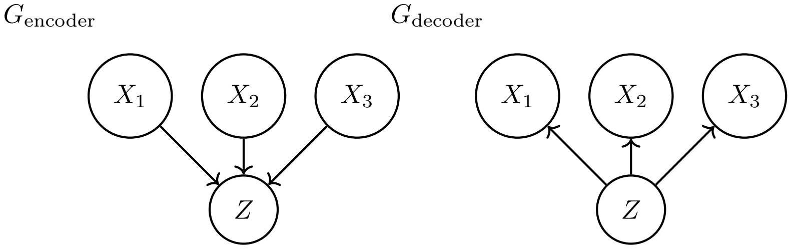


Figure 15: Encoder and decoder graphs for a single shared latent representation UMVIB

All views X_1, X_2, and X_3 are compressed into a single latent variable Z, which is then used to reconstruct each view. The model is illustrated in Figure 15, and the corresponding loss function is:

$$L_{\text{UMVIB}_1} = \tilde{I}^E(X_1, X_2, X_3; Z) - \beta(\tilde{I}^D(X_1; Z) + \tilde{I}^D(X_2; Z) + \tilde{I}^D(X_3; Z)). \tag{43}$$

Using the same library of terms as before:

$$
\begin{aligned}
L_{\mathrm{UMVIB}_1} \approx{}& \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z|x_{1i}, x_{2i}, x_{3i})\|r(z)) \\
&- \beta\left(\frac{1}{N}\sum_{i=1}^{N}\int dz p(z|x_{1i}, x_{2i}, x_{3i})\ln(q(x_{1i}|z)) + \frac{1}{N}\sum_{i=1}^{N}\int dz p(z|x_{1i}, x_{2i}, x_{3i})\ln(q(x_{2i}|z))\right. \\
&\left. + \frac{1}{N}\sum_{i=1}^{N}\int dz p(z|x_{1i}, x_{2i}, x_{3i})\ln(q(x_{3i}|z))\right). \qquad (44)
\end{aligned}
$$

- **Shared and private latent representations (UMVIB$_2$):**

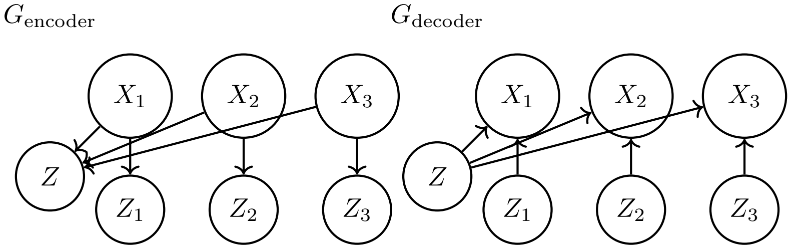


Figure 16: Encoder and decoder graphs for shared and private latent representations UMVIB

An alternative formulation introduces both a shared latent variable Z and private latent variables Z_1, Z_2, and Z_3, corresponding to each view. This structure extends DVCCA-private to three variables. Each view X_i is reconstructed from its private latent representation Z_i along with the shared latent variable Z. The model is illustrated in Fig. 16, and the corresponding loss function is:

$$
\begin{aligned}
L_{\mathrm{UMVIB}_2} ={}& \tilde{I}^E((X_1, X_2, X_3); Z) + \tilde{I}^E(X_1; Z_1) + \tilde{I}^E(X_2; Z_2) + \tilde{I}^E(X_3; Z_3) \\
&- \beta(\tilde{I}^D(X_1; (Z, Z_1)) + \tilde{I}^D(X_2; (Z, Z_2)) + \tilde{I}^D(X_3; (Z, Z_3))). \qquad (45)
\end{aligned}
$$

Using the same library of terms as before, this becomes:

$$
\begin{aligned}
L_{\mathrm{UMVIB}_2} \approx{}& \frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z|x_{1i}, x_{2i}, x_{3i})\|r(z)) + \sum_{j=1}^{3}\frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(z_j|x_{j_i})\|r_j(z)) \\
&- \beta\left(\frac{1}{N}\sum_{i=1}^{N}\int dz dz_1 dz_2 dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{1i}|z, z_1))\right. \\
&+ \frac{1}{N}\sum_{i=1}^{N}\int dz dz_1 dz_2 dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{2i}|z, z_2)) \\
&\left. + \frac{1}{N}\sum_{i=1}^{N}\int dz dz_1 dz_2 dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{3i}|z, z_3))\right). \qquad (46)
\end{aligned}
$$

B.3.2 Supervised Multi-View IB

Several approaches in the literature, such as Lee and Van der Schaar (2021); Wang et al. (2019); Huang et al. (2022); Grazioli et al. (2022), focus on compressing multiple views or modalities X_i into one or more latent variables Z (or Z and Z_i) and uses Z to predict a supervisory signal/label Y (or in addition to using both Z and Z_i to reconstruct X_i as well). Within this framework, we can also consider two primary formulations:

- **Single shared latent representation (SMVIB$_1$):**

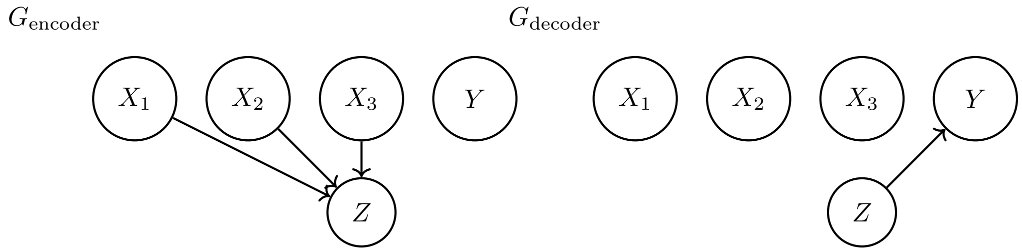


Figure 17: Encoder and decoder graphs for a single shared latent representation SMVIB

All views X_1, X_2, and X_3 are compressed into a single latent variable Z, which is then used to predict the supervising signal Y. The model is illustrated in Figure 17, and the corresponding loss function is:

$$L_{\text{SVMIB}_1} = \tilde{I}^E(X_1, X_2, X_3; Z) - \beta(\tilde{I}^D(Y; Z)). \tag{47}$$

Using the same library of terms as before, we get:

$$L_{\text{SMVIB}_1} \approx \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z|x_{1i}, x_{2i}, x_{3i}) \| r(z)) - \beta \left(\frac{1}{N}\sum_{i=1}^{N} \int dz p(z|x_{1i}, x_{2i}, x_{3i}) \ln(q(y_i|z)) \right). \tag{48}$$

- **Shared and private latent representations (SMVIB$_2$):**

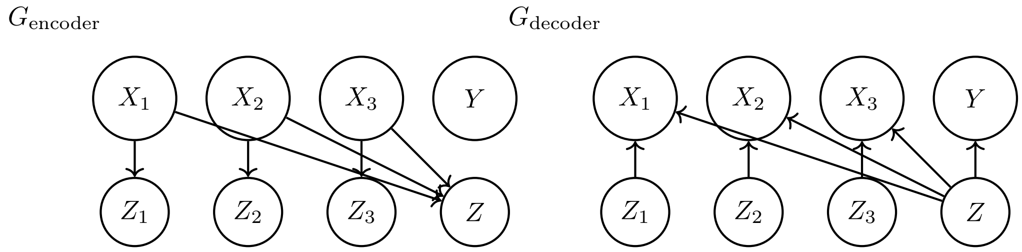


Figure 18: Encoder and decoder graphs for shared and private latent representations SMVIB

An alternative formulation introduces both a shared latent variable Z and private latent variables Z_1, Z_2, and Z_3, corresponding to each view. The shared latent variable Z reconstructs the supervising signal Y. In addition, each view X_i is reconstructed from its private latent representation Z_i along with the shared latent variable Z. The model is illustrated in Figure 18, and the corresponding loss function is:

$$\begin{aligned} L_{\text{SMVIB}_2} = {} & \tilde{I}^E((X_1, X_2, X_3); Z) + \tilde{I}^E(X_1; Z_1) + \tilde{I}^E(X_2; Z_2) + \tilde{I}^E(X_3; Z_3) \\ & - \beta(\tilde{I}^D(X_1; (Z, Z_1)) + \tilde{I}^D(X_2; (Z, Z_2)) + \tilde{I}^D(X_3; (Z, Z_3)) + \tilde{I}^D(Z; Y)). \end{aligned} \tag{49}$$

Using the same library of terms as before, this becomes:

$$
\begin{aligned}
L_{\text{SMVIB}_2} \approx \frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z|x_{1i}, x_{2i}, x_{3i})\|r(z)) + \sum_{j=1}^{3}\frac{1}{N}\sum_{i=1}^{N} D_{\text{KL}}(p(z_j|x_{j_i})\|r_j(z))\\
-\beta\Big(\frac{1}{N}\sum_{i=1}^{N}\int dzdz_1dz_2dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{1i}|z, z_1))\\
+\frac{1}{N}\sum_{i=1}^{N}\int dzdz_1dz_2dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{2i}|z, z_2))\\
+\frac{1}{N}\sum_{i=1}^{N}\int dzdz_1dz_2dz_3 p(z|x_{1i}, x_{2i}, x_{3i})p(z_1|x_{1i})p(z_2|x_{2i})p(z_3|x_{3i})\ln(q(x_{3i}|z, z_3))\\
+\frac{1}{N}\sum_{i=1}^{N}\int dzdyp(z|x_{1i}, x_{2i}, x_{3i})p(y_i|z)\Big). \qquad (50)
\end{aligned}
$$

B.3.3 Discussion

There exist many other structures that have been explored in the multi-view representation learning literature, including conditional VIB (Shi et al., 2019; Hwang et al., 2021), which is formulated in terms of conditional information. These types of structures are beyond the current scope of our framework. However, they could be represented by an encoder mapping from all independent views X_ν to Z, subtracted from another encoder mapping from the joint view $\vec{X}$ to Z. Coupled with this would be a decoder mapping from Z to the independent views X_ν (or the joint view $\vec{X}$, analogous to the Joint-DVCCA). Similarly, one can use our framework to represent other multi-view approaches, or their approximations (Lee and Van der Schaar, 2021; Wan et al., 2021; Hwang et al., 2021). This underscores the breadth of methods seeking to address specific questions by exploring known or assumed statistical dependencies within data, and also the generality of our approach, which can re-derive these methods.

B.4 Widely used state of the art methods

B.4.1 Barlow Twins - BT

For Barlow Twins (Zbontar et al., 2021), two different views X and Y are compressed using the same deterministic compression function $\mu(\cdot)$ into Z_X and Z_Y respectively ($p(z_x|x) = \delta(z_x - \mu(x))$ and $p(z_y|y) = \delta(z_y - \mu(y))$). In training, one optimizes the cross-correlation matrix, C, between Z_X and Z_Y to be as close to identity as possible. This is equivalent to maximizing the information between $I(Z_X, Z_Y)$ when Z_X and Z_Y are Gaussian. This has the same loss structure as the deterministic symmetric information bottleneck with no reconstruction (Eq 15), but with a shared deterministic encoder for both Z_X and Z_Y.

$$L_{\text{BT}} = -\tilde{I}^D(Z_X; Z_Y) = \frac{1}{2}(\ln(\det(I - CC^T))). \qquad (51)$$

This has the same minimum as:

$$L_{\text{BT}} = \sum_i (1 - C_{ii})^2 + \lambda \sum_{i,j} C_{ij}^2. \qquad (52)$$

B.4.2 Contrastive Language-Image Pretraining - CLIP

Contrastive Language-Image Pretraining (Radford et al., 2021) was introduced to learn visual concepts from natural language supervision. CLIP simultaneously compresses both the text and the images into a common latent space by maximizing a similarity metric between the compression found for the image and the text. Crucially, it also penalizes the similarity between samples of images and text that are not paired with one another. The CLIP loss is

$$L_{\text{CLIP}} = \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_j}/T\right)}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_j}\cdot\vec{z}_{y_i}/T\right)}\right) \tag{53}$$

where $\vec{z}_{x_i} = \vec{\mu}_{Z_X}(x_i)$, $\vec{z}_{y_i} = \vec{\mu}_{Z_Y}(y_i)$ and the functions $\vec{\mu}_{Z_X}$ and $\vec{\mu}_{Z_Y}$ are parameterized by networks.

We show now that L_{CLIP} is related to the Symmetric Information Bottleneck, but with deterministic encoders for both Z_X and Z_Y and in the $\beta \to \infty$ limit. Specifically, CLIP's compression of the views X and Y amounts to using deterministic encoder functions into Z_X and Z_Y, respectively, such that $p(z_x|x) = \delta(\vec{z}_x - \vec{\mu}_{Z_X}(x))$ and $p(z_y|y) = \delta(\vec{z}_y - \vec{\mu}_{Z_Y}(y))$. Below we show that, if one represents $p(z_x, z_y) = \frac{\exp\left(\vec{z_x}\cdot\vec{z_y}/T\right)}{Z}$, then, in the limit $N \to \infty$, $L_{\text{CLIP}} \to \frac{1}{2}(-I(Z_X; Z_Y) + H(Z_X, Z_Y))$, where I and H are estimated from samples. Further, if one instead interprets $p(z_x|z_y) = p(z_y|z_x) = \frac{\exp\left(\vec{z_x}\cdot\vec{z_y}/T\right)}{Z}$, then $L_{\text{CLIP}} \to -I(Z_X; Z_Y)$ in the same $N \to \infty$ limit. In both of these limits, L_{CLIP} has the same structure as the deterministic symmetric information bottleneck with no reconstruction (Eq. 15), but in the first case, there is an extra differential entropy term, $H(Z_X, Z_Y)$.

In the expressions below, we use the limit sign, →, to indicate convergence for $N \to \infty$. That is, by using the limit, we replace the average over a sample with the expectation value. We first start with the case of $p(z_x, z_y) = \frac{\exp\left(\vec{z_x}\cdot\vec{z_y}/T\right)}{Z}$. Then:

$$\begin{aligned}
L_{\text{CLIP}} &= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_j}/T\right)}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_j}\cdot\vec{z}_{y_i}/T\right)}\right)\\
&= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}, z_{y_i})}{\frac{1}{N}\sum_{j=1}^{N}p(z_{x_i}, z_{y_j})}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}, z_{y_i})}{\frac{1}{N}\sum_{j=1}^{N}p(z_{x_j}, z_{y_i})}\right)\\
&\to \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}, z_{y_i})}{p(z_{x_i})}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}, z_{y_i})}{p(z_{y_i})}\right)\\
&= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p^2(z_{x_i}, z_{y_i})}{p(z_{x_i})p(z_{y_i})}\right) = \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}, z_{y_i})}{p(z_{x_i})p(z_{y_i})}\right) - \frac{1}{2N}\sum_{i=1}^{N}\ln\left(p(z_{x_i}, z_{y_i})\right)\\
&\to \frac{1}{2}(-I(Z_X, Z_Y) + H(Z_X, Z_Y)),
\end{aligned} \tag{54}$$

proving the assertion.

Now consider the case $p(z_y|z_x) = p(z_x|z_y) = \frac{\exp\left(\vec{z}_x\cdot\vec{z}_y/T\right)}{Z}$. Then:

$$\begin{aligned}
L_{\text{CLIP}} &= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_j}/T\right)}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{\exp\left(\vec{z}_{x_i}\cdot\vec{z}_{y_i}/T\right)}{\frac{1}{N}\sum_{j=1}^{N}\exp\left(\vec{z}_{x_j}\cdot\vec{z}_{y_i}/T\right)}\right)\\
&= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}|z_{y_i})}{\frac{1}{N}\sum_{j=1}^{N}p(z_{x_i}|z_{y_j})}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{y_i}|z_{x_i})}{\frac{1}{N}\sum_{j=1}^{N}p(z_{y_i}|z_{x_j})}\right)\\
&\to \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}|z_{y_i})}{\int p(z_{x_i}|z_y)p(z_y)dz_y}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{y_i}|z_{x_i})}{\int p(z_{y_i}|z_x)p(z_x)dz_x}\right)\\
&= \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{x_i}|z_{y_i})}{p(z_{x_i})}\right) + \frac{-1}{2N}\sum_{i=1}^{N}\ln\left(\frac{p(z_{y_i}|z_{x_i})}{p(z_{y_i})}\right)\\
&\to \frac{-1}{2}I(Z_X; Z_Y) + \frac{-1}{2}I(Z_X; Z_Y) = -I(Z_X; Z_Y),
\end{aligned} \tag{55}$$

which completes the proof.

We end this section by discussing these two interpretations of the CLIP loss function (the original work by Radford et al. (2021) did not offer an interpretation).

For the first interpretation, we note that combining mutual information and entropy in the same loss function is an unconventional choice. Indeed, for continuous variables, mutual information is dimensionless and reparameterization invariant. In contrast, differential entropy has units of $\ln p$, and it depends on the parameterization of the variables. One cannot argue with the exceptional success of CLIP on real-world data, and yet one must question whether restoring the dimensionless structure of the loss by subtracting the empirically sampled version of the differential entropy term from Eq. (53) would lead to even better results.

For the second interpretation, we note that the assumption $p(z_x|z_y) = p(z_y|z_x)$ is very restrictive. It amounts to requiring either (i) a near-identity mapping between z_x and z_y, or (ii) that $p(z_x) = p(z_y)$ for all admissible values of z_x and z_y. The latter case is equivalent to saying that the marginal distributions are uniform, and yet the joint is not.

In view of these arguments, it would be interesting to explore if replacing the CLIP loss function with an empirically sampled version of the mutual information, $I(Z_X; Z_Y)$—that is, making CLIP exactly an SIB problem—would result in better performance. It would also be interesting to explore whether the embeddings learned by CLIP on real-world data result in satisfying $p(z_x|z_y) = p(z_y|z_x)$. More generally, it is interesting to explore whether there is another interpretation of $\frac{\exp(\vec{z}_x \cdot \vec{z}_y / T)}{Z}$ that would result in a more meaningful limit of the CLIP loss function L_{CLIP} for $N \to \infty$.

Appendix C. Additional MNIST Results

In this section, we present supplementary results derived from the methods in Tables 1&4.

C.1 Additional results tables for the best parameters

We report classification accuracy using SVM on data X (SVM on Y is in the main text Tbl. 2), and using neural networks on both X and Y.

Table 6: Maximum accuracy from a linear SVM and the optimal k_Z and β for variational DR methods on the X dataset. († fixed values)

Method	**Acc. %**	$k_{Z\,\mathbf{best}}$	95% $k_{Z\,\mathbf{range}}$	$\boldsymbol{\beta}_{\text{best}}$	95% $\beta_{\mathbf{range}}$
Baseline	57.8	$784^\dagger$	-	-	-
PCA	58.0	256	[32,265*]	-	-
CCA	54.4	256	[8,265*]	-	-
β-VAE	84.4	256	[128,265*]	4	[2,8]
DVIB	87.3	128	[4,265*]	512	[8,1024*]
DVCCA	86.1	256	[64,265*]	$1^\dagger$	-
β-DVCCA	88.9	256	[128,265*]	4	[1,128]
DVCCA-private	85.3	128	[32,265*]	$1^\dagger$	-
β-DVCCA-private	85.3	128	[32,265*]	1	[1,8]
DVSIB	**92.9**	256	[64,265*]	256	[4,1024*]
DVSIB-private	**92.6**	256	[**32**,265*]	128	[8,1024*]

Table 7: Maximum accuracy from a feed forward neural network and the optimal k_Z and β for variational DR methods on the Y and the joined $[X, Y]$ datasets. († fixed values)

Method	**Acc. %**	$k_{Z\textbf{best}}$	95% $k_{Z\textbf{range}}$	$\boldsymbol{\beta}_{\text{best}}$	95% $\beta_{\textbf{range}}$
Baseline	92.8	$784^\dagger$	-	-	-
PCA	97.6	128	[16,256*]	-	-
CCA	90.2	256	[32,256*]	-	-
β-VAE	**98.4**	64	[8,256*]	64	[2,1024*]
DVIB	90.4	128	[8,256*]	1024	[8,1024*]
DVCCA	91.3	16	[4,256*]	$1^\dagger$	-
β-DVCCA	97.5	128	[8,256*]	512	[2,1024*]
DVCCA-private	93.8	16	[2,256*]	$1^\dagger$	-
β-DVCCA-private	97.5	256	[**2**,256*]	32	[1,1024*]
DVSIB	**98.3**	256	[**4**,256*]	32	[2,1024*]
DVSIB-private	**98.3**	256	[**4**,256*]	32	[2,1024*]
Baseline-joint	97.7	$1568^\dagger$	-	-	-
joint-DVCCA	93.7	256	[8,256*]	$1^\dagger$	-
β-joint-DVCCA	**98.9**	64	[8,256*]	512	[2,1024*]
joint-DVCCA-private	93.5	16	[4,256*]	$1^\dagger$	-
β-joint-DVCCA-private	95.6	32	[4,256*]	512	[1,1024*]

Table 8: Maximum accuracy from a neural network the optimal k_Z and β for variational DR methods on the X dataset. († fixed values)

Method	**Acc. %**	$k_{Z\textbf{best}}$	95% $k_{Z\textbf{range}}$	$\boldsymbol{\beta}_{\text{best}}$	95% $\beta_{\textbf{range}}$
Baseline	92.8	$784^\dagger$	-	-	-
PCA	91.9	64	[32,256*]	-	-
CCA	72.6	256	[256,256*]	-	-
β-VAE	93.3	256	[16,256*]	256	[2,1024*]
DVIB	87.5	4	[2,256*]	1024	[4,1024*]
DVCCA	87.5	128	[8,256*]	$1^\dagger$	-
β-DVCCA	92.2	64	[8,256*]	32	[2,1024*]
DVCCA-private	88.2	8	[8,256*]	$1^\dagger$	-
β-DVCCA-private	90.7	256	[4,256*]	8	[1,1024*]
DVSIB	**93.9**	128	[**8**,256*]	16	[2,1024*]
DVSIB-private	92.8	32	[8,256*]	256	[4,1024*]

C.2 t-SNE embeddings at best parameters

Figure 19 display 2d t-SNE embeddings for variables Z_X and Z_Y generated by various considered DR methods.

C.3 t-SNE embeddings at $k_{Z_X} = k_{Z_Y} = 2$

We now demonstrate how different DR methods behave when the compressed variables are restricted to have not more than 2 dimensions, cf. Fig. 20.

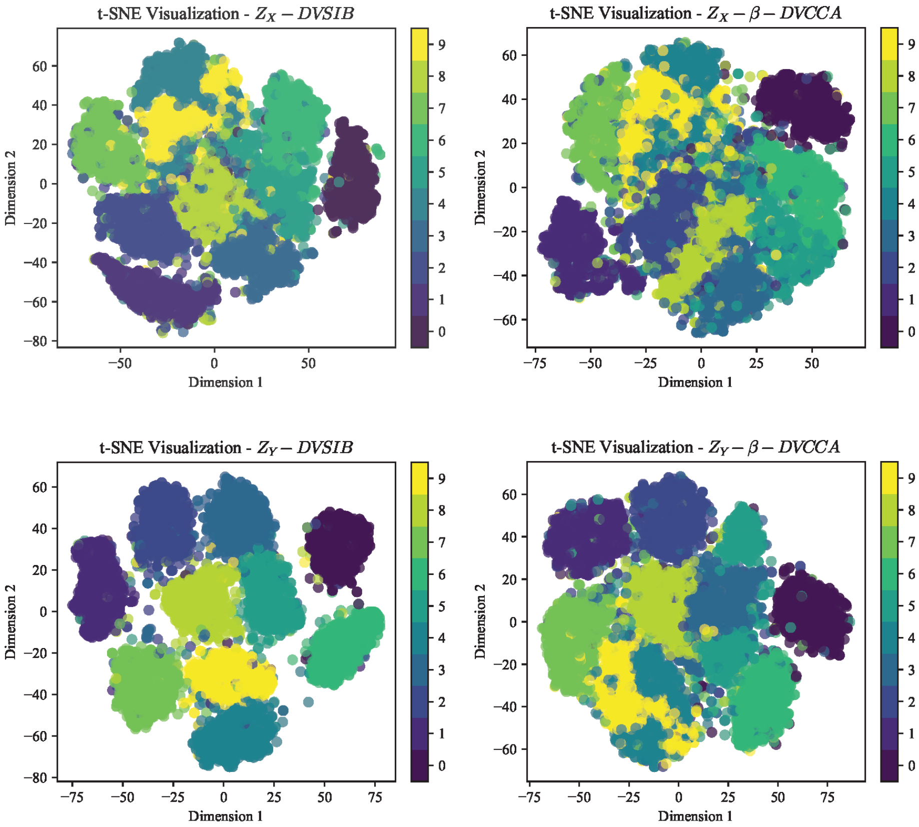


Figure 19: Clustering of embeddings for best β, k_{Z_X} (top) k_{Z_Y} (bottom) for DVSIB and β-DVCCA.

C.4 DVSIB-private embeddings

C.4.1 At best parameters

Figure 21 shows the t-SNE embeddings of the private latent variables constructed by DVSIB-private, colored by the digit label. To the extent that the labels do not cluster, private latent variables do not preserve the label information shared between X and Y.

C.4.2 At $k_{Z_X} = k_{Z_Y} = 2$

Figure 22 shows the embeddings constructed by DVSIB-private, colored by the digit label, rotations, scales, and noise factors for X and Y. Private latent variables at 2 latent dimensions preserve a little about the label information shared between X and Y, but clearly preserve the scale information for X, even at only two latent dimensions.

C.5 Testing Training Efficiency

We tested an SVM's classification accuracy for distinguishing digits based on latent subspaces created by DVSIB, β-VAE, CCA, and PCA trained using different amounts of samples. Figure 5 in the main text shows the results for 60 epochs of training with latent spaces of dimension $k_{Z_X} = k_{Z_Y} = 64$. The DVSIB and β-VAE were trained with $\beta = 1024$. Figure 23 shows the SVM's classification accuracy

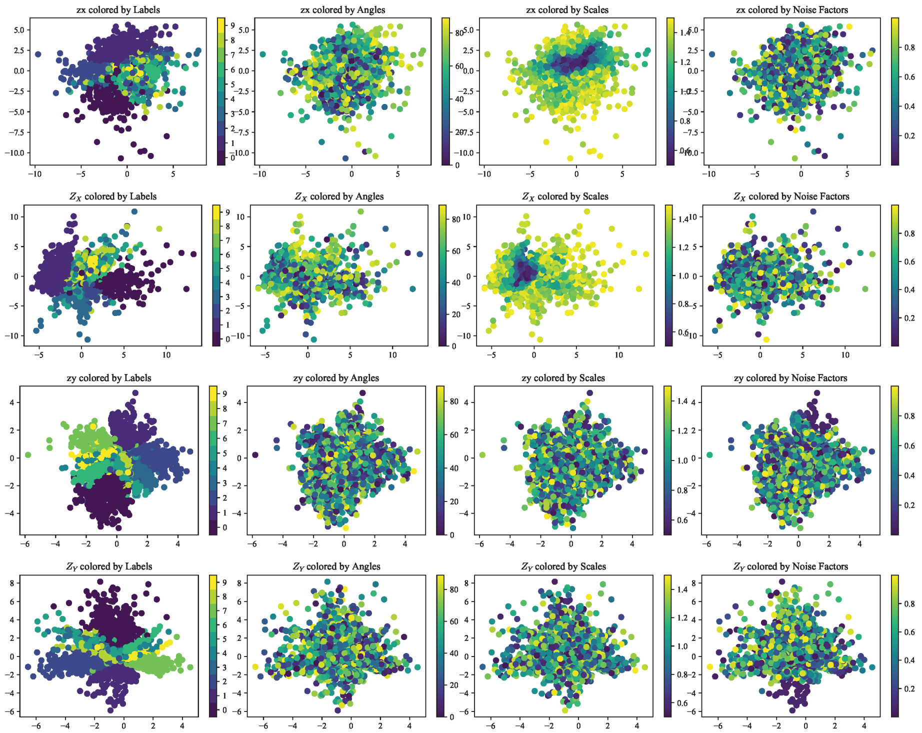


Figure 20: Clustering of embeddings when restricting $k_Z = 2$ for DVSIB and β-DVCCA colored by labels, rotations, scales, and noise factors, respectively. The top two rows show results for the X dataset, while the bottom two rows correspond to the Y dataset.

for a range of latent dimensions (from right to left): $k_{Z_X} = k_{Z_Y} = 2, 16, 64, 256$. Additionally, it shows the results for different amounts of training time for the encoders ranging from 20 epochs (top row) to 100 epochs (bottom row). As explained in the main text, we plot a log-log graph of $100 - A$ versus $1/n$. Plotted in this way, high accuracy appears at the bottom, and large sample sizes are at the left of the plots. DVSIB, β-VAE, and CCA often appear linear when plotted this way, implying that they follow the form $A = 100 - c/n^m$. Steeper slopes m on these plots correspond to a faster increase in the accuracy with the sample size. This parameter sweep shows that the tested methods have not had time to fully converge at low epoch numbers. Additionally, increasing the number of latent dimensions helps the SVMs untangle the non-linearities present in the data and improves the corresponding classifiers.

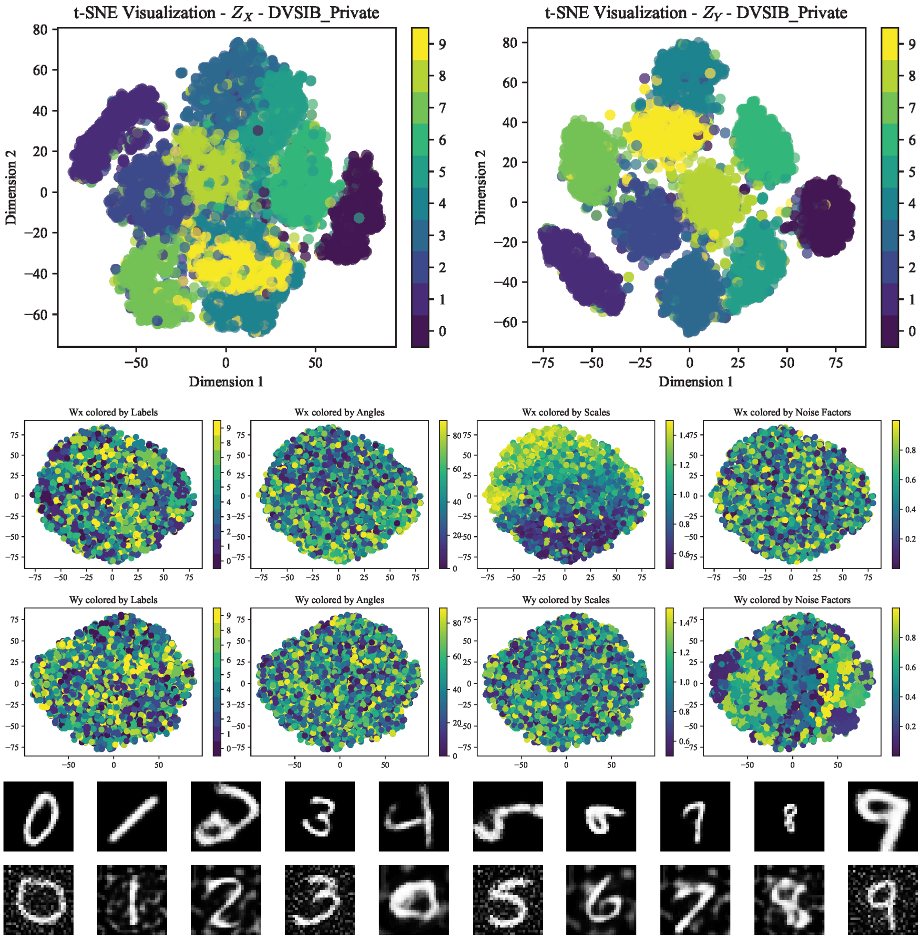


Figure 21: DVSIB-private t-SNE embeddings at best k_Z and β: Top row shows Z_X (left) and Z_Y (right). Second and third rows display W_X and W_Y, colored by labels, rotations, scales, and noise factors, respectively. Bottom row presents digit reconstructions using both shared and private information, demonstrating how private components capture variations in background, scaling, and rotation.

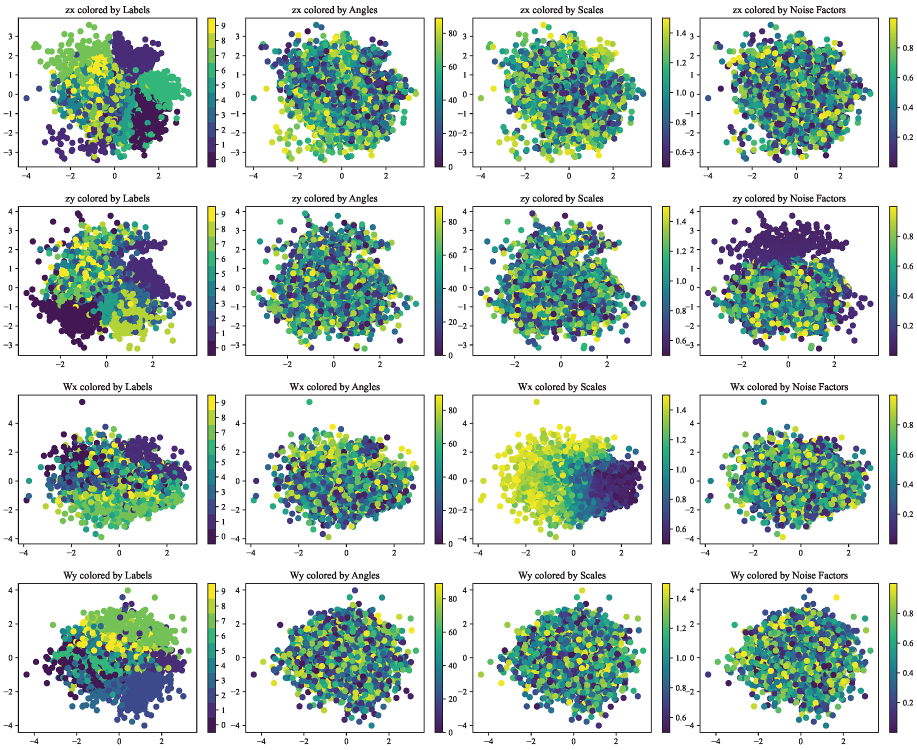


Figure 22: DVSIB-private embeddings: The first two rows correspond to Z_X and Z_Y, respectively, while the third and fourth rows show W_X and W_Y. Each row is colored by labels, rotations, scales, and noise factors, respectively.

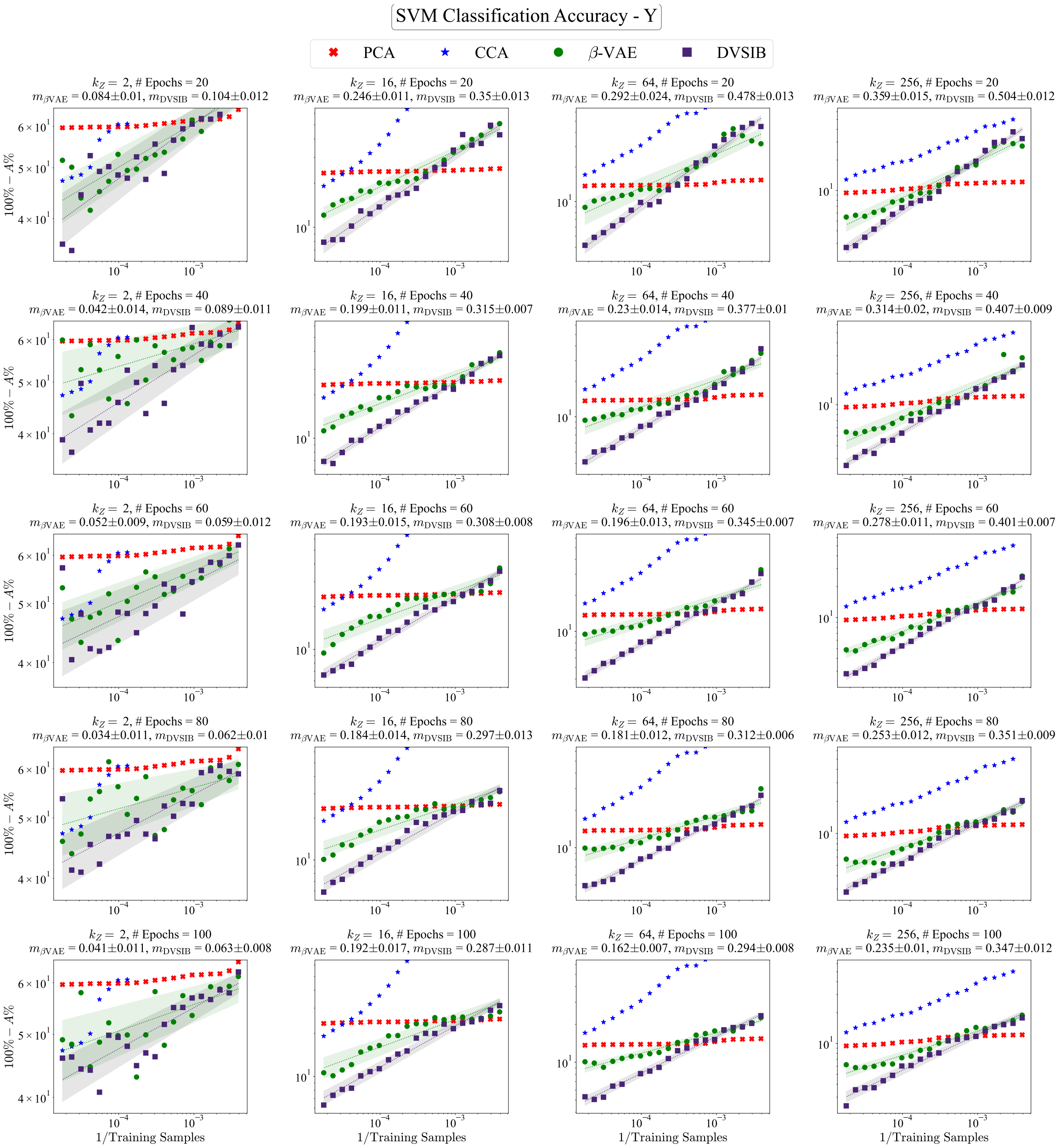


Figure 23: Log-log plot of $100 - A$ vs $1/n$. DVSIB has a steeper slope than β-VAE corresponding to faster convergence with fewer samples for DVSIB. Plots vary $k_Z = 2, 16, 64, 256$ and training epochs $20, 40, 60, 80, 100$.

Appendix D. Convolutional DVSIB

D.1 Noisy CIFAR-100 dataset

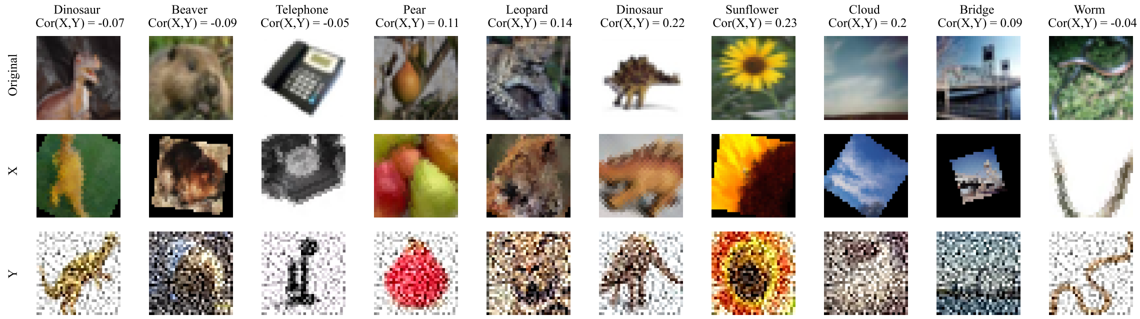


Figure 24: Examples from the Noisy CIFAR-100 dataset. Each column shows instances from a random category from CIFAR-100. For each category, we show an example of an unperturbed image of that class (top), a rotated and scaled example image from the class X (middle), and a noised example image from the class with Perlin background noise Y (bottom). The correlation between the X and Y views is calculated for the shown pairs.

We evaluate different DR methods on an adaptation of the CIFAR-100 dataset (Krizhevsky, 2009). CIFAR-100 consists of $60,000$ color images in 100 classes, with 600 images per class. Each image is of size 32×32 pixels and has 3 color channels (RGB), corresponding to a total number of 3072 pixels.

To create the Noisy CIFAR-100 dataset, we generate two distinct views of each image class. The first view (X) is the original image, which is randomly rotated by an angle uniformly sampled between 0 and $\frac{\pi}{2}$ and scaled by a factor uniformly distributed between 0.5 and 1.5. The second view (Y) is derived from the original image but with an added background Perlin noise (Perlin, 1985), where the noise factor is independently sampled for each RGB channel from a uniform distribution between 0 and 1. Both views are rescaled to have pixel intensities in the range $[0, 1)$.

The dataset was shuffled within labels, retaining only the shared label identity between the two transformed views (X and Y). The dataset is partitioned into training (80%), testing (10%), and validation (10%) subsets.

Figure 24 illustrates some representative examples from different categories in the Noisy CIFAR-100 dataset. For each shown category, we show an original image, a transformed image from X (rotated and scaled), and a noisy image from Y (with Perlin noise).

D.2 Architecture

Since the Noisy CIFAR-100 dataset is more complex, and to showcase the flexibility of our framework with diverse data modalities, we extend our model to incorporate convolutional neural networks (CNNs) for the encoder and decoder architectures. Unlike the fully connected feed-forward networks used for Noisy MNIST, the convolutional structures are better suited for handling images with multiple channels.

We use convolutional encoders and decoders as detailed in the D.2.1 & D.2.2 for the Variational Autoencoder (VAE), β-DVCCA, DVCCA ($\beta = 1$), and DVSIB models. While Conv-VAE is well established in the literature (Rezende et al., 2014; Pu et al., 2016; Kingma et al., 2014), we introduce novel adaptations: Conv-β-DVCCA and Conv-DVSIB. Additionally, we enhance DVSIB by integrating a clipped version of the MINE estimator, known as SMILE (Song and Ermon, 2019).

This modified estimator, set with a clipping factor $\tau = 5$, is described in detail in the D.2.3. The SMILE estimator improves the estimation stability and accuracy of mutual information, leading to enhanced performance of Conv-DVSIB.

D.2.1 Encoder $I^E(X; Z_X)$

The convolutional encoder architecture is designed to handle the $32 \times 32 \times 3$ image input, using three convolutional layers. The first layer has 64 filters, followed by batch normalization and ReLU activation, followed by a max-pooling layer to reduce the spatial dimensions. The second convolutional layer has 128 filters, also followed by batch normalization, ReLU activation, and max-pooling. The final convolutional layer consists of 256 filters, further reducing the image dimensions with max-pooling. All the convolutional layers have a kernel size of 3×3 and padding of 1, with the max-pooling layers having a stride of 2

The flattened output of the convolutional layers is passed through a fully connected layer of size 1024 with ReLU activation and dropout of 0.5 for regularization, then another fully connected linear layer of size 1024. Then the 1024 dimensional representation is mapped to two output layers that generate the mean (μ) and log variance ($\log \sigma^2$) for the latent space. The fully connected layers are initialized with Xavier initialization (Glorot and Bengio, 2010) for stability.

D.2.2 Decoder $I^D(X; Z_X)$

The convolutional decoder mirrors this structure in reverse. Starting from the latent representation, it expands the data through two fully connected layers of size 1024, reshaping it to a $256 \times 4 \times 4$ tensor. The deconvolutional layers upsample the feature maps back to the original image size using transpose convolutions, with batch normalization and ReLU activation, and a final Sigmoid activation to ensure pixel intensities lie in the range $[0, 1]$.

D.2.3 Mutual Information Estimator $I_{\text{SMILE}}(Z_X; Z_Y)$

To enhance the mutual information estimation in Conv-DVSIB, we employ the SMILE estimator (Song and Ermon, 2019), explained in more detail in Appx. A.3.

The SMILE estimator is implemented using a concatenated critic with three hidden layers, each containing 256 neurons. The input layer takes the joint low-dimensional representations from both views, $[Z_X, Z_Y]$, and maps them to a single-neuron output layer. This adaptation significantly stabilizes the learning of mutual information and contributes to improved performance in the Conv-DVSIB model.

D.3 Results - Tables

Table 10 summarizes the linear SVM classification accuracy for each method tested: Conv-VAE, Conv-β-DVCCA, Conv-DVCCA ($\beta = 1$), and Conv-DVSIB. Since Noisy CIFAR-100 is a more challenging multi-class dataset compared to Noisy MNIST, and we have *simple* implementations for the encoders and decoders, we report the top-k accuracies for $k = 1, 5, 10, 20$, where top-k refers to the fraction of instances where the true class is among the top-k predicted classes. The table reports the test accuracies, using a separate dataset that was not seen during training or validation. The values of k_Z shown are the ones that gave the highest accuracy for each method after sweeping over different k_Z values. We found that $\beta = 1024$ consistently gave the best results for all methods (Sec. D.4). The results indicate that Conv-DVSIB outperforms all other methods across different top-k accuracies. In addition, our newly improved method Conv-β-DVCCA clearly outperforms the original Conv-DVCCA keeping all other parameters fixed. Note that this is not meant to compete with the best possible accuracy for the Noisy CIFAR-100 algorithms which is beyond the message of this paper, but rather to show the comparable behavior of all the methods on equal footage. A better comparison moving towards the state-of-the-art level is in Sec.E.

D.3.1 SVM on X

Table 9: Linear SVM classification accuracy on Noisy CIFAR (X data). We report the top-k accuracies for $k = 1, 5, 10, 20$, along with the corresponding k_Z values that provided the highest accuracies for each method. The methods tested are Conv-VAE, Conv-β-DVCCA, Conv-DVCCA ($\beta = 1$), and Conv-DVSIB. The best results are shown in bold.

Method	Top 1	k_Z	Top 5	k_Z	Top 10	k_Z	Top 20	k_Z
Conv-VAE	8.43	128	24.67	128	36.97	128	54.28	128
Conv-DVCCA	0.72	2	3.72	2	7.85	2	16.55	2
Conv-β-DVCCA	10.65	128	30.17	128	42.83	128	57.83	128
Conv-DVSIB	**14.72**	128	**38.23**	128	**52.58**	16	**68.48**	16

D.3.2 SVM on Y

Table 10: Linear SVM classification accuracy on Noisy CIFAR (Y data). We report the top-k accuracies for $k = 1, 5, 10, 20$, along with the corresponding k_Z values that provided the highest accuracies for each method. The methods tested are Conv-VAE, Conv-β-DVCCA, Conv-DVCCA ($\beta = 1$), and Conv-DVSIB. The best results are shown in bold.

Method	Top 1	k_Z	Top 5	k_Z	Top 10	k_Z	Top 20	k_Z
Conv-VAE	13.93	64	35.97	64	49.6	64	65.25	64
Conv-DVCCA	0.72	2	3.72	2	7.85	2	16.55	2
Conv-β-DVCCA	15.07	128	35.63	128	48.33	128	64.2	64
Conv-DVSIB	**23.75**	128	**52.15**	32	**65.13**	32	**78.93**	32

D.4 Results - Figures

In the supplementary figures, we further analyze the impact of k_Z and β on the performance:

1. Figure 25 shows the effect of varying k_Z on top-k accuracy, with β value chosen to maximize the reported accuracy. The x-axis represents different values of k_Z, while the four columns display the top-1, top-5, top-10, and top-20 accuracies. The first row shows results for the X data, while the second row shows results for the Y data.

2. Figure 26 illustrates the effect of varying β with k_Z value chosen to maximize the reported accuracy. The x-axis represents different values of β, and the columns correspond to the different top-k accuracies as described above.

D.4.1 k_Z sweeps

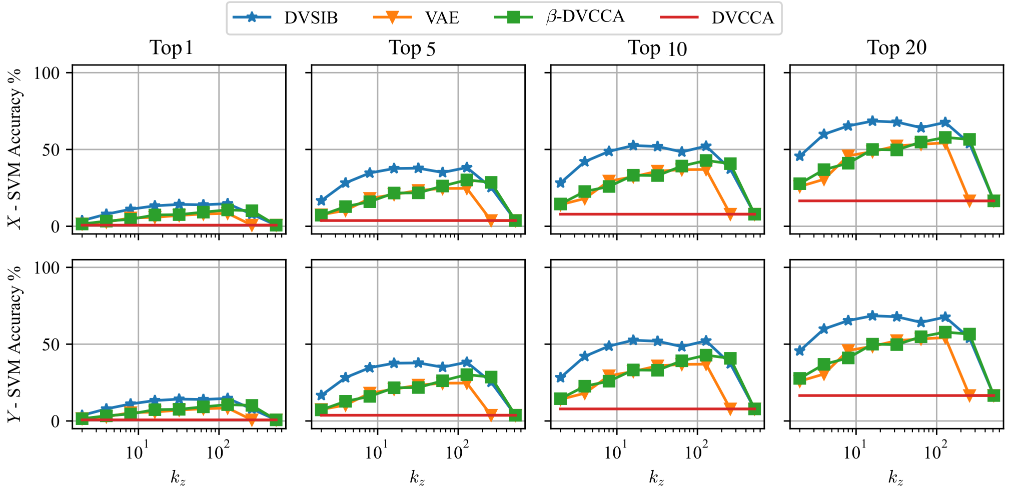


Figure 25: Top-k classification accuracies (Top-1, Top-5, Top-10, Top-20) as a function of k_Z for Noisy CIFAR. The x-axis shows different values of k_Z, and β is fixed at the highest value tested ($\beta = 1024$). The first row shows results for X data, while the second row shows results for Y data. We observe a saturation or decline in accuracy beyond $k_Z = 128$.

D.4.2 β sweeps

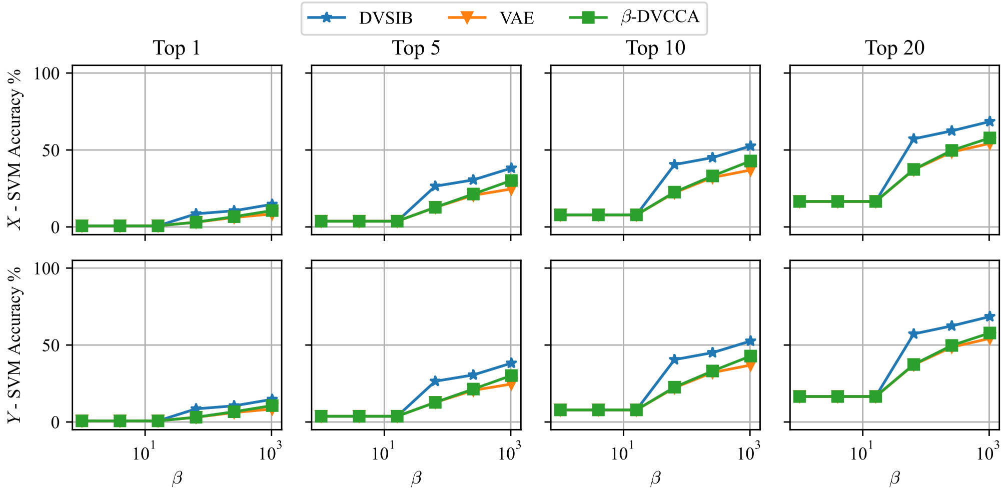


Figure 26: Top-k classification accuracies (Top-1, Top-5, Top-10, Top-20) as a function of β for Noisy CIFAR. The x-axis shows different values of β, with k_Z fixed at the highest value tested ($k_Z = 128$). The first row shows results for X data, while the second row shows results for Y data. We notice an increase in accuracy as β increases up to 1024.

Appendix E. ResNet-DVSIB

E.1 Barlow Twins and Mixed Barlow Twins

Self-supervised learning (SSL) methods such as Barlow Twins (Zbontar et al., 2021) have demonstrated remarkable performance in representation learning tasks. However, a key limitation of the standard Barlow Twins (BT) approach is its tendency to overfit the training data, particularly when trained on small or moderate-sized datasets. To address this issue, Bandara et al. (2023) introduced a mixup-based regularization technique that significantly improves generalization by encouraging better alignment between augmented views of the same data.

The Barlow Twins objective is formulated as:

$$\mathcal{L}_{BT} = \sum_i (1 - C_{ii})^2 + \lambda \sum_i \sum_{j \neq i} C_{ij}^2, \tag{56}$$

where C is the cross-correlation matrix of the normalized embeddings from two different augmentations of the same image, and λ controls the weight of the off-diagonal terms.

To further improve generalization, Bandara et al. (2023) proposed a mixup-based regularization. Given an image batch $\mathbf{x}$, they generate two augmentations $\mathbf{y}_a$ and $\mathbf{y}_b$ and compute their corresponding embeddings z_a and z_b using a ResNet (18 or 50)[3] (He et al., 2016) backbone. A mixed input is constructed as:

$$\mathbf{y}_m = \alpha \mathbf{y}_a + (1 - \alpha)\mathbf{y}_b^{\pi}, \tag{57}$$

where $\alpha \sim \text{Beta}(1.0, 1.0)$ and π is a random permutation of batch indices. The mixed embedding z_m is then computed using the same encoder network. The regularization loss is computed based on the difference between the estimated and ground-truth cross-correlation matrices:

$$\mathcal{L}_{mix} = \lambda_{mix} \lambda \left(\sum_{i,j} (C_m^a - C_m^{a,gt})^2 + \sum_{i,j} (C_m^b - C_m^{b,gt})^2 \right), \tag{58}$$

where C_m^a and C_m^b are the cross-correlation matrices for the mixed embeddings with respect to z_a and z_b, and $C_m^{a,gt}$ and $C_m^{b,gt}$ are their corresponding ground-truth values.

The total loss is then:

$$\mathcal{L} = \mathcal{L}_{BT} + \mathcal{L}_{mix}. \tag{59}$$

E.2 ResNet-DVSIB: Adaptation for Variational Information Bottleneck

We extend the DVSIB architecture (Sec. 2.1) by incorporating elements of the Barlow Twins approach and comparing it to Barlow Twins Zbontar et al. (2021) and the Mixed Barlow Twins Bandara et al. (2023).

E.2.1 Encoder $I^E(X; Z_X)$

The encoder follows the ResNet-18 backbone up to the 512-dimensional embedding layer, followed by a modified projection head, then a fully connected layer mapping (512, 512). The layer is followed by a Batch normalization and a ReLU activation. Then followed by two parallel output layers of size $(512, k_{Z_\mu})$ and $(512, k_{Z_{\log \sigma^2}})$ to produce the mean and log-variance of the variational posterior.

E.2.2 Decoder $I^D(X; Z_X)$

The decoder reconstructs images from a lower-dimensional feature representation and is designed to mirror a ResNet-18 encoder. It consists of a decoder-projection head and a transposed convolutional

[3] For all of this section, we use the ResNet-18 backbone

decoder. The decoder-projection head is a two-layer fully connected network that projects the embedded dimensionality feature vector ($\mathbb{R}^{k_Z}$) to match the ResNet 18 encoder's final output dimension ($\mathbb{R}^{512}$), with batch normalization and ReLU activation applied after each layer. The transposed convolutional decoder then progressively upsamples the feature map from $(512, 1, 1)$ to $(3, 32, 32)$ through a sequence of six transposed convolutional layers, each followed by batch normalization and ReLU activation, except for the final output layer, with proper padding and strides.

E.2.3 Mutual Information Estimator $I_{\text{InfoNCE}}(Z_X; Z_Y)$

The mutual information term is estimated using the InfoNCE estimator with a separable critic (Oord et al., 2018). The critic consists of two networks, g and h, each with a fully connected layer mapping: $(k_Z, 512)$ with Xavier uniform initialization (Glorot and Bengio, 2010), and a RelU activation followed by a fully connected final layer of $(512, 8192)$, also with a Xavier uniform initialization. These networks compute $g(Z_X)$ and $h(Z_Y)$, which are then combined via a dot product to form a separable critic function $f(Z_X, Z_Y) = g(X) \cdot h(Y)$ for InfoNCE. The estimator is explained in more detail in Appx. A.3

E.3 Training Setup and Results

To make our ResNet-DVSIB architecture resemble BT, we enforce both encoding terms, $I^E(X; Z_X)$ and $I^E(Y; Z_Y)$, to share the same encoder. Similarly, the decoder is also shared for both views. The representations Z_X and Z_Y are then obtained by passing the two augmented versions, X and Y, through this common encoder network[4].

We initialize the encoder portion of ResNet-DVSIB (before the projection head) using the pre-trained model from Bandara et al. (2023) trained on CIFAR-100 at a k_Z of 1024, where the reported top-1 accuracy is 61%[5]. We replace the projection head with our variational head at $k_z = 128$, while initializing the decoder and mutual information estimator from scratch.

During training, and because the encoder portion is already a good starting point, we set its loss weight to zero ($\mathcal{L} = 0 * I^E - \beta I^D$) but allow it to train indirectly through the other terms I^D in the loss, but with a learning rate of $0.1\times$ the base learning rate. The full model is optimized using cosine annealing with an initial learning rate of 10^{-3}, running for 1000 epochs with early stopping after 100 epochs of no improvement. Training stabilizes around 50 epochs, with a highest top-1 accuracy of 58.6% (top-5: 84.28%) achieved around 500 epochs, as shown in Fig. 27.

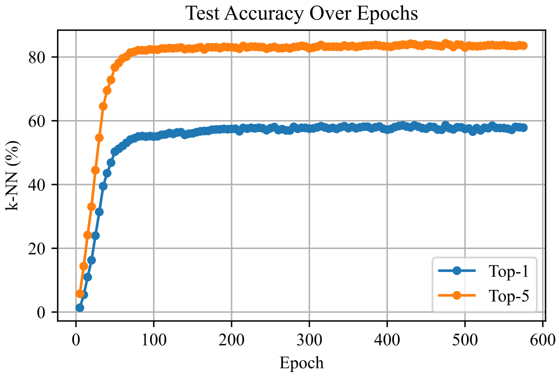


Figure 27: k-NN classification accuracy over training for top-1 and top-5 on the ResNet-DVSIB model using the CIFAR-100 dataset. The figure shows saturation around 50 epochs, with minor fluctuations afterward.

[4] More generally, this setup can be applied in DVSIB when we want to ensure that the same features are learned from different inputs.

[5] Saved model and code of Bandara et al. (2023) available at `https://github.com/wgcban/mix-bt/tree/main`. We adapted parts of this code to integrate with ours and made the necessary modifications to align it with our ResNet-DVSIB setup

Noting that Bandara et al. (2023) report the top-1 accuracy of the original BT on the CIFAR-100 dataset to be around 50% for $k_Z = 128$ and around 58% for $k_Z = 1024$, while the mixed BT reports around 60% and 61% for $k_Z = 128, 1024$ respectively[6].

E.4 Discussion

Our results show that the adapted DVSIB framework achieves comparable performance to the 1024-dimensional BT model while using only 128 latent dimensions. Additionally, our model not only produces useful representations, but also learns a latent distribution and includes a decoder, allowing for further interpretability. Although no exhaustive tuning was performed, these results demonstrate the viability of the DVSIB method, and the DVMIB framework in general, in achieving near-the-state-of-the-art performance in self-supervised learning.

[6] Refer to Figure 3 of Bandara et al. (2023) for BT and Figure 4 for mixed BT. Note that the only explicitly reported value is for the mixed BT model at $k_Z = 1024$, which was 61%, while other values were inferred from the graph.

References

Eslam Abdelaleem, Ahmed Roman, K. Michael Martini, and Ilya Nemenman. Simultaneous dimensionality reduction: A data efficient approach for multimodal representations learning. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=Ni14fXbyTV`.

Eslam Abdelaleem, K Michael Martini, and Ilya Nemenman. Accurate estimation of mutual information in high dimensional data. *arXiv preprint arXiv:2506.00330*, 2025.

Alex Alemi, Ian Fischer, Josh Dillon, and Kevin Murphy. Deep variational information bottleneck. In *ICLR*, 2017.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1247–1255, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning*, pages 1298–1312. PMLR, 2022.

Wele Gedara Chaminda Bandara, Celso M De Melo, and Vishal M Patel. Guarding barlow twins against overfitting with mixed samples. *arXiv preprint arXiv:2312.02151*, 2023.

Feng Bao. Disentangled variational information bottleneck for multiview representation learning. In *Artificial Intelligence: First CAAI International Conference, CICAI 2021, Hangzhou, China, June 5–6, 2021, Proceedings, Part II 1*, pages 91–102. Springer, 2021.

Adrien Bardes, Jean Ponce, and Yann Lecun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR 2022-International Conference on Learning Representations*, 2022.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pages 531–540. PMLR, 2018.

Adrian Benton, Huda Khayrallah, Biman Gujral, Dee Ann Reisinger, Sheng Zhang, and Raman Arora. Deep generalized canonical correlation analysis. *arXiv preprint arXiv:1702.02519*, 2017.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

Sarath Chandar, Mitesh M. Khapra, Hugo Larochelle, and Balaraman Ravindran. Correlational neural networks. *Neural Computation*, 28(2):257–285, 2016. doi: 10.1162/NECO_a_00801.

Gal Chechik, Amir Globerson, Naftali Tishby, and Yair Weiss. Information bottleneck for gaussian variables. *Advances in Neural Information Processing Systems*, 16, 2003.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

Kenneth Clark, Bruce Vendt, Kirk Smith, John Freymann, Justin Kirby, Paul Koppel, Stephen Moore, Stanley Phillips, David Maffitt, Michael Pringle, et al. The cancer imaging archive (tcia): maintaining and operating a public information repository. *Journal of digital imaging*, 26:1045–1057, 2013.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *8th International Conference on Learning Representations*. OpenReview. net, 2020a.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. *arXiv preprint arXiv:2002.07017*, 2020b.

Nir Friedman, Ori Mosenzon, Noam Slonim, and Naftali Tishby. Multivariate information bottleneck. *arXiv preprint arXiv:1301.2270*, 2013.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Filippo Grazioli, Raman Siarheyeu, Israa Alqassem, Andreas Henschel, Giampaolo Pileggi, and Andrea Meiser. Microbiome-based disease prediction with multimodal variational information bottlenecks. *PLOS Computational Biology*, 18(4):e1010050, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2016.

Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15, 2002.

Geoffrey E Hinton and Ruslan Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. doi: 10.1126/science.1127647.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 1936. doi: 10.1007/978-1-4612-4380-9_14.

Shizhe Hu, Zenglin Shi, and Yangdong Ye. Dmib: Dual-correlated multivariate information bottleneck for multiview clustering. *IEEE Transactions on Cybernetics*, 52(6):4260–4274, 2020.

Teng-Hui Huang, Aly El Gamal, and Hesham El Gamal. On the multi-view information bottleneck representation. In *2022 IEEE Information Theory Workshop (ITW)*, pages 37–42. IEEE, 2022.

Rachael P Huntley, Tony Sawford, Prudence Mutowo-Meullenet, Aleksandra Shypitsyna, Carlos Bonilla, Maria J Martin, and Claire O'Donovan. The goa database: gene ontology annotation updates for 2015. *Nucleic acids research*, 43(D1):D1057–D1063, 2015.

HyeongJoo Hwang, Geon-Hyeong Kim, Seunghoon Hong, and Kee-Eung Kim. Multi-view representation learning via total correlation objective. *Advances in Neural Information Processing Systems*, 34:12194–12207, 2021.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

Mahdi Karami and Dale Schuurmans. Deep probabilistic canonical correlation analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(9):8055–8063, May 2021. doi: 10.1609/aaai.v35i9.16982.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27, 2014.

John W Krakauer, Asif A Ghazanfar, Alex Gomez-Marin, Malcolm A MacIver, and David Poeppel. Neuroscience needs behavior: correcting a reductionist bias. *Neuron*, 93(3):480–490, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Changhee Lee and Mihaela Van der Schaar. A variational information bottleneck approach to multi-omics data integration. In *International Conference on Artificial Intelligence and Statistics*, pages 1513–1521. PMLR, 2021.

Marco Lorenzi, Andre Altmann, Boris Gutman, Selina Wray, Charles Arber, Derrek P Hibar, Neda Jahanshad, Jonathan M Schott, Daniel C Alexander, Paul M Thompson, et al. Susceptibility of brain atrophy to trib3 in alzheimer's disease, evidence from functional prioritization in imaging genetics. *Proceedings of the National Academy of Sciences*, 115(12):3162–3167, 2018.

K Michael Martini and Ilya Nemenman. Data efficiency, dimensionality reduction, and the generalized symmetric information bottleneck. *arXiv preprint arXiv:2309.05649*, 2023.

Rui Meng, Tianyi Luo, and Kristofer Bouchard. Compressed predictive information coding. *arXiv preprint arXiv:2203.02051*, 2022.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Rich Pang, Benjamin J Lansdell, and Adrienne L Fairhall. Dimensionality reduction in neuroscience. *Current Biology*, 26(14):R656–R660, 2016.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3):287–296, 1985.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5171–5180. PMLR, 09–15 Jun 2019.

Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29, 2016.

Lin Qiu, Vernon M Chinchilli, and Lin Lin. Variational interpretable deep canonical correlation analysis. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Chris Pal, Hugo Larochelle, Aaron Courville, and Bernt Schiele. Movie description. *International Journal of Computer Vision*, 2017.

Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. How2: a large-scale dataset for multimodal language understanding. *arXiv preprint arXiv:1811.00347*, 2018.

Yuge Shi, Brooks Paige, Philip Torr, et al. Variational mixture-of-experts autoencoders for multi-modal deep generative models. *Advances in neural information processing systems*, 32, 2019.

Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv preprint arXiv:1910.06222*, 2019.

Nicholas A Steinmetz, Cagatay Aydin, Anna Lebedeva, Michael Okun, Marius Pachitariu, Marius Bauza, Maxime Beau, Jai Bhagat, Claudia Böhm, Martijn Broux, et al. Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, 372(6539):eabf4588, 2021.

Milan Studenỳ and Jirina Vejnarová. The multiinformation function as a tool for measuring stochastic dependence. *Learning in graphical models*, pages 261–297, 1998.

Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. Joint multimodal learning with deep generative models. *arXiv preprint arXiv:1611.01891*, 2016.

Valentine Svensson, Roser Vento-Tormo, and Sarah A Teichmann. Exponential scaling of single-cell rna-seq in the past decade. *Nature protocols*, 13(4):599–604, 2018.

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

Anne E Urai, Brent Doiron, Andrew M Leifer, and Anne K Churchland. Large-scale neural recordings call for new insights to link brain and behavior. *Nature neuroscience*, 25(1):11–19, 2022.

Hrishikesh D Vinod. Canonical ridge and econometrics of joint production. *Journal of Econometrics*, 1976. doi: 10.1016/0304-4076(76)90010-5.

Zhibin Wan, Changqing Zhang, Pengfei Zhu, and Qinghua Hu. Multi-view information-bottleneck representation learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 10085–10092, 2021.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

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*, pages 37–45. SIAM, 2019.

Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *International conference on machine learning*, pages 1083–1092. PMLR, 2015.

Weiran Wang, Xinchen Yan2 Honglak Lee, and Karen Livescu. Deep variational canonical correlation analysis. *arXiv preprint arXiv:1610.03454*, 2016.

Svante Wold, Michael Sjöström, and Lennart Eriksson. Pls-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58(2):109–130, 2001. ISSN 0169-7439. doi: https://doi.org/10.1016/S0169-7439(01)00155-1.

Hok Shing Wong, Li Wang, Raymond Chan, and Tieyong Zeng. Deep tensor cca for multi-view learning. *IEEE Transactions on Big Data*, 8(6):1664–1677, 2021.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive learning of medical visual representations from paired images and text. In *Machine Learning for Healthcare Conference*, pages 2–25. PMLR, 2022.

Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):14049, 2017.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=ydopy-e6Dg`.

Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Finn Årup Nielsen, Lars Kai Hansen, and Stephen C Strother. Canonical ridge analysis with ridge parameter optimization. *NeuroImage*, 1998. doi: 10.1016/s1053-8119(18)31591-x.