StrWAEs to Invariant Representations

Hyunjong Lee [*1] **Yedarm Seong** [*2] **Sungdong Lee** [3] **Joong-Ho Won** [1 2]

Abstract

Autoencoders have become an indispensable tool for generative modeling and representation learning in high dimensions. Imposing structural constraints such as conditional independence in order to capture invariance of latent variables to nuisance information has been attempted through adding *ad hoc* penalties to the loss function mostly in the variational autoencoder (VAE) context, often based on heuristics. This paper demonstrates that Wasserstein autoencoders (WAEs) are highly flexible in embracing such structural constraints. Well-known extensions of VAEs for this purpose are gracefully handled within the framework of WAEs. In particular, given a conditional independence structure of the generative model (decoder), corresponding encoder structure and penalties are *derived* from the functional constraints that define the WAE. These structural uses of WAEs, termed StrWAEs ("stairways"), open up a principled way of penalizing autoencoders to impose structural constraints. Utilizing these advantages, we present a handful of results on semi-supervised classification, conditional generation, and invariant representation tasks.

1. Introduction

The ability to learn informative representations of data with minimal supervision is a key challenge in machine learning (Tschannen et al., 2018). To address this challenge, autoencoders have become an indispensable toolkit. An autoencoder consists of the encoder, which maps the input to a low-dimensional representation, and the decoder, that maps a representation back to a reconstruction of the input. Thus an autoencoder can be considered a nonlinear factor analysis model as the latent variable provided by the encoder carries the meaning of "representation" and the decoder can be used for generative modeling of the input data distribution. Most autoencoders can be formulated as minimizing some "distance" between the distribution P_X of input random variable X and the distribution $g_\sharp P_Z$ of the reconstruction $G = g(Z)$, where Z is the latent variable or representation having distribution P_Z, g is either deterministic or probabilistic decoder, and $\sharp$ is the push-forward operator (in the latter case g is read as the conditional distribution of G given Z), which is variationally described in terms of an encoder $Q_{Z|X}$. For instance, the variational autoencoder (VAE, Kingma & Welling, 2014) minimizes

$$D_{\text{VAE}}(P_X, g_\sharp P_Z) = \inf_{Q_{Z|X} \in \mathcal{Q}} \mathbb{E}_{P_X}[\mathcal{D}_{KL}(Q_{Z|X} \| P_Z) - \mathbb{E}_{Q_{Z|X}} \log g(Z)] \quad (1)$$

over the set of probabilistic decoders or conditional densities g of G given Z, where $\mathcal{D}_{KL}$ is the Kullback-Leibler (KL) divergence, and the Wasserstein autoencoder (WAE, Tolstikhin et al., 2018) minimizes

$$D_{\text{WAE}}(P_X, g_\sharp P_Z) = \inf_{Q_{Z|X} \in \mathcal{Q}} \mathbb{E}_{P_X} \mathbb{E}_{Q_{Z|X}} d^p(X, g(Z)) \quad (2)$$

over the set of deterministic decoders g, where d is the metric in the space of input X and $p \geq 1$. Set $\mathcal{Q}$ restricts the search space for the encoder. In VAEs, a popular choice is a class of normal distributions

$$\mathcal{Q} = \{Q_{Z|\{X=x\}} = N(\mu(x), \text{diag}(\Sigma(x))) \mid \mu, \Sigma \in \mathcal{NN}\}$$

where $\mathcal{NN}$ is a class of functions parametrized by neural networks. In WAEs, the choice

$$\mathcal{Q} = \{Q_{Z|X} \mid Q_Z := \mathbb{E}_{P_X} Q_{Z|X} = P_Z\} \quad (3)$$

is considered; Q_Z is called an *aggregate posterior* of Z.

Of course, the notion of "informativeness" depends on the downstream task. The variation in the observations that are not relevant to the particular task is often called "nuisance" and is desirable to be suppressed from the representation. For example, in semi-supervised learning where the goal is to use labeled as well as unlabeled data to perform learning tasks such as digit generation (Van Engelen & Hoos, 2020),

[*]Equal contribution [1]Department of Statistics, Seoul National University, Seoul, Korea [2]Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul, Korea [3]Department of Medicine, Yong Loo Lin School of Medicine, National University of Singapore, Singapore. Correspondence to: Joong-Ho Won <wonj@stats.snu.ac.kr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

and learning representations separated from class specification performs well (Kingma et al., 2014); in obtaining representations of facial images, those that are invariant to attributes (such as lighting conditions, poses, or presence of eyeglasses) are often sought. A popular approach to this goal is to explicitly separate informative and nuisance variables in the generative model by factorization. This approach imposes a structure on the decoder. Additionally the encoder is further factorized and a penalty promoting *invariance* of the encoded representation to the nuisance variable can be added. A well-known example is the HSIC-constrained VAE (HCV, Lopez et al., 2018), in which a decoder with a nuisance variable is factorized in a similar way to the semi-supervised VAE (Kingma et al., 2014). A resembling factorization of the encoder (variational posterior) is *assumed*, and penalties promoting this factorization, such as the Hilbert-Schmidt Independence Criterion (HSIC, Gretton et al., 2007) are added to the evidence lower bound (ELBO). The Characteristic Capturing VAE (CCVAE, Joy et al., 2021) splits the latent space and only certain elements of the latent variable are associated with the nuisance variables. Another example is the Fader Networks (Lample et al., 2018), in which the deterministic decoder takes an additional input of the attribute and an adversarial penalty that hinders the accurate prediction of the attribute by the deterministic, unfactorized encoder.

These examples illustrate that, while the generative model (decoder structure) can be chosen suitably for the downstream task, *there is no principled way of imposing the corresponding encoder structure*. The goal of this paper is to show that the WAE framework allows us to *automatically determine* the encoder structure corresponding to the imposed decoder structure. Specifically, when the deterministic decoder g in (2) is structured to handle the conditional independence relations in the imposed generative model, then the constraint set (amounting to the $\mathcal{Q}$ in (3)) that makes the left-hand side of (2) a proper (power of) Wasserstein distance determines the factorization of the (deterministic) encoder. In practice, the hard constraints in $\mathcal{Q}$ are relaxed and (2) is solved in a penalized form. Another benefit of the WAE framework is that the cited constraint set can be *systematically translated to penalties*. Therefore, in addition to the theoretical advantage that the penalized form equals a genuine distributional distance for sufficiently large penalty parameter while that of (1) remains a lower bound of the negative log-likelihood of the model, the *ad hoc* manner of designing penalties for the assumed encoder structure prevalent in the VAE literature can be avoided in the WAE framework. Further, nuisance variables are treated in a unified fashion and the downstream tasks are not limited to semi-supervised classification.

After providing necessary background in Section 2, we explain how the WAE framework leads to structured encoders given a generative model through examples reflecting downstream tasks in Section 3. We would call these structured uses of WAEs the *Structured Wasserstein AutoEncoders*, abbreviated as StrWAEs (pronounced "stairways"). Various instances of StrWAEs are experimented in Section 4 for various datasets in tasks such as semi-supervised classification, conditional generation, style transfer, attribute manipulation, and obtaining invariant representations. Several implications of our framework are discussed in Section 5.

2. Preliminaries

In fitting a given probability distribution P_X of a random variable X on a measurable space $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$, where $\mathcal{X} \subset \mathbb{R}^D$ equipped with metric d, by a generative model distribution P_G of sample G on the same meaurable space, one may consider minimizing the (p-th power of) p-Wasserstein distance between the two distributions, i.e.,

$$\min_{P_G \in \mathcal{M}} \left\{ W_p^p(P_X, P_G) := \inf_{\pi \in \mathcal{P}(P_X, P_G)} \mathbb{E}_\pi \, d^p(X, G) \right\}.$$

Here, $\mathcal{M}$ is the model space of probability distributions, $\mathcal{P}(P_X, P_G)$ is the coupling or the set of all joint distributions on $(\mathcal{X} \times \mathcal{X}, \mathcal{B}(\mathcal{X} \times \mathcal{X}))$ having marginals P_X and P_G. Often the sample G is generated by transforming a variable in a latent space. When $G \overset{\text{a.s.}}{=} g(Z)$ for a latent variable Z in a probability space $(\mathcal{Z}, \mathcal{B}(\mathcal{Z}), P_Z)$, $\mathcal{Z} \subset \mathbb{R}^l$ (typically $l \ll D$), and measurable function g, then P_G is denoted by $g_\sharp P_Z$. In this setting, as discussed in Section 1, Tolstikhin et al. (2018) show that $W_p^p(P_X, g_\sharp P_Z) = D_{\text{WAE}}(P_X, g_\sharp P_Z)$ (see (2)), with the constraint set given in (3). It is further claimed by Patrini et al. (2020, Theorem A.1) that the set of probabilistic encoders $Q_{Z|X}$ can be reduced to be deterministic, i.e., $Z \overset{\text{a.s.}}{=} f(X)$ for f measurable. However, their proof relies on the existence of a right inverse $\tilde{g}$ of g when the codomain of the latter is restricted to its range. Unfortunately, it is incorrectly stated that $(\tilde{g} \circ g)_\sharp P_Z(A)$ is equal to $P_Z(\tilde{g}^{-1}(g^{-1}(A))$, whereas the correct statement is $(\tilde{g} \circ g)_\sharp P_Z(A) = P_Z(g^{-1}(\tilde{g}^{-1}(A)))$. From the latter, the key equality $(\tilde{g} \circ T)_\sharp P_X = P_Z$, where T is the optimal transport map, does not hold. To this end, if $\tilde{g}$ is a *left* inverse of g, the desired equality holds. The reason that a right inverse is needed in Patrini et al. (2020) is to ensure that $g \circ (\tilde{g} \circ T) = T$ almost surely in P_X and treat $\tilde{g} \circ T$ as the sought deterministic encoder f. To achieve this goal with a left inverse, a more involved argument is needed. The proof of the following result can be found in Appendix A.

Theorem 2.1. *If P_X is not atomic, a cost function c is continuous, and the measurable function $g : \mathcal{Z} \to \mathcal{X}$ is injective, then*

$$W_c(P_X, g_\sharp P_Z) = \inf_{f \in \mathcal{Q}} \mathbb{E}_{P_X} \, c(X, g(f(X))), \tag{4}$$

where $\mathcal{Q}$ is the set of all measurable functions from $\mathcal{X}$ to $\mathcal{Z}$

such that $f_\sharp P_X = P_Z$.

Remark 2.2. Injectivity of decoder g can be imposed by using convolutional layers with leaky ReLU activation function; g is injective if the convolution kernels form bases of the corresponding vector spaces, which occurs almost surely if they are initialized randomly (Puthawala et al., 2022).

In practice the set $\mathcal{Q}$ can be relaxed to $\mathcal{F}$, a class of all measurable functions paremeterized by deep neural networks, which contains a minimizer of the right-hand side of (4); the constraint $f_\sharp P_X = P_Z$ can be met by adding a penalty $\lambda \mathcal{D}(f_\sharp P_X \| P_Z)$ for sufficiently large multiplier $\lambda > 0$ and a divergence $\mathcal{D}$ between two distributions. Thus the generative modeling problem based on p-Wasserstein distance can be formulated as

$$\inf_{g \in \mathcal{G}} \inf_{f \in \mathcal{F}} \{\mathbb{E}_{P_X} d^p(X, g(f(X))) + \lambda \mathcal{D}(f_\sharp P_X \| P_Z)\}, \quad (5)$$

for $\mathcal{G}$ a set of injective measurable functions from $\mathcal{Z}$ to $\mathcal{X}$, typically parameterized by deep neural networks. The function $f : \mathcal{X} \to \mathcal{Z}$ has an interpretation of an *encoder* and $g : \mathcal{Z} \to \mathcal{X}$ has an interpretation of a *decoder*.

3. Learning invariant representations with StrWAEs

In this section, we illustrate how a WAE is fully specified by a model of decoder. Consider the graphical models of Figure 1a, which we call the chain model. As we see in Remark 3.3 and Section 3, the chain model is flexible and captures many of the decoder structures considered in the VAE literature (Kingma et al., 2014; Louizos et al., 2016; Lopez et al., 2018; Joy et al., 2021; Feng et al., 2021). Variables $Y_i \in \mathcal{Y}_i \subset \mathbb{R}^{k_i}$ for $i = 1, \dots, n$ represent the observed nuisance variables but $Y_n \in \mathbb{R}^{k_n}$ may be partially observed. The $Z_i \in \mathcal{Z}_i \subset \mathbb{R}^{l_i}$ for $i = 1, \dots, n$ is a latent variable carrying information of $Y_{i+1}, \dots, Y_n$ about G with which we want to mimic the observable variable X. Here, $Y_1, \dots, Y_n$ are assumed to be independent. Notice that all functions are measurable.

3.1. Formulating the Wasserstein Distance as Structured Autoencoder

Notation. First, we define some notation for readability.

- $\mathcal{A}_0 : h \mapsto (\cdot, h(\cdot))$
- $\mathcal{A}_1 : f \mapsto ((x, s, y) \mapsto (s, y, f(x, s, y)))$, i.e., $(\mathcal{A}_1 f)(x, s, y) = (s, y, f(x, s, y))$; s can be omitted.
- $(f_1, f_2) : x \mapsto (f_1(x), f_2(x))$
- $Y_{i:j} = (Y_i, \dots, Y_j)$, $\mathcal{Y}_{i:j} = \Pi_{k=i}^j \mathcal{Y}_k$ for $i \le j$
- The joint distribution $P_{XY_{1:n}}$ of X and $Y_{1:n}$ has marginals $P_X, P_{Y_1}, \dots, P_{Y_n}$

The operators $\mathcal{A}_i$ are related to the graph of the function, and the subscript indicates how many elements to fix when drawing the graph.

Complete data. For the chain model, we derive simple autoencoder forms of the Wasserstein distance between the data distribution and the generative model following the approach in Theorem 2.1. The chain model well-defines the joint distribution $P_{GY_{1:n}} = P_{G|Y_{1:n}} P_{Y_{1:n}} = (g(Y_{1:n}, \cdot)_\sharp P_{Z_n}) P_{Y_{1:n}}$ where $g(y_{1:n}, z) := g_1(Y_1, \cdots, g_n(Y_n, z))$. However, since the decoder g depends both on the observable $Y_{1:n}$ and latent variable Z_n, whether an equivalent of Theorem 2.1 holds for the chain model remains a question. The following theorem answers this question affirmatively. When all the input variables are observed, the encoder that takes as inputs the observed labels $Y_{1:n}$ in addition to X and the distributional constraints are naturally derived during formulation.

Theorem 3.1. *If the conditional distributions $P_{X|Y_{1:n}}$ are non-atomic and $g_1, \dots, g_n$ are injective, then*

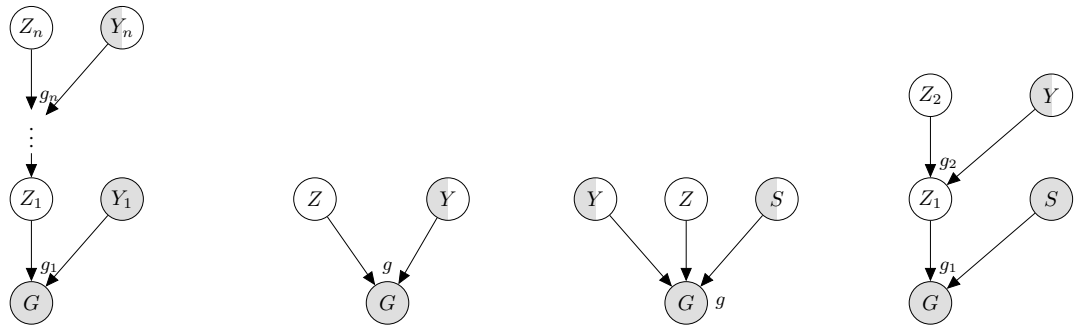
$$W_p^p(P_{XY_{1:n}}, P_{GY_{1:n}}) = \inf_{f \in \mathcal{F}} \mathbb{E}_{P_{XY_{1:n}}} d^p(X, g \circ (\mathcal{A}_1 f)(X, Y_{1:n})), \quad (6)$$

where $\mathcal{F} = \{f : \mathcal{X} \times \mathcal{Y}_{1:n} \to \mathcal{Z}_n \mid (\mathcal{A}_1 f)_\sharp P_{XY_{1:n}} = (\bigotimes_{i=1}^n P_{Y_i}) \otimes P_{Z_n}\}$.

The proof is given in Appendix A.2. This result states that the encoder f should satisfy the constraint $f(X, Y_{1:n}) \overset{d}{=} Z_n$ and $f(X, Y_{1:n}) \perp\!\!\!\perp Y_{1:n}$, where $\overset{d}{=}$ denotes equality in distribution. The intuition behind Theorem 3.1 is as follows. In unstructured WAEs, in order to find the optimal transport between the distribution P_X and the decoded one $g(Z)$, the aggregated posterior $f_\sharp P_X$ should match perfectly the prior distribution P_Z. If there is a observed nuisance variable $Y \equiv Y_{1:n}$ and the decoder takes the form of $g(Y, Z_n)$, then the *joint* distributions of the pairs (X, Y) and $(g(Y, Z_n), Y)$ need to be close. Within the WAE framework, proceeding as above, the encoder should take both X and Y as input and the distribution of $f(X, Y)$ should match perfectly that of the latent variable Z_n, i.e., $f(X, Y) \overset{d}{=} Z_n$. Since Y and Z_n are independent by construction, so should be Y and $f(X, Y)$, i.e., $f(X, Y) \perp\!\!\!\perp Y$. We also emphasize that the functional form of the encoder *mirrors* that of the decoder.

Missing observations. If Y_n is missing, then only $X, Y_1, \dots, Y_{n-1}$ are observable hence we can estimate the Wasserstein distance between $P_{XY_{1:(n-1)}}$ and $P_{GY_{1:(n-1)}}$. Theorem 3.1 applies directly if we replace P_{Z_n} with $P_{Y_n} \otimes P_{Z_n}$:

$$W_p^p(P_{XY_{1:(n-1)}}, P_{GY_{1:(n-1)}}) = \inf_{(h, \breve{h}) \in \mathcal{H}} \mathbb{E}_{P_{XY_{1:(n-1)}}} \big[d^p(X, g \circ (\mathcal{A}_1(h, \breve{h}))(X, Y_{1:(n-1)}))\big], \quad (7)$$



(a) Chain model (b) M2 model (c) M2 + nuisance (d) stacked model

Figure 1: Examples of generative models for StrWAEs. Gray nodes (resp. white nodes) represent the observable (resp. missing) variables. Half-filled nodes indicate that they are partially observable. (b), (c), and (d) are special cases of (a).

where $\mathcal{H} = \{(h, \breve{h}) | h : \mathcal{X} \times \mathcal{Y}_{1:(n-1)} \to \mathcal{Y}_n, \breve{h} : \mathcal{X} \times \mathcal{Y}_{1:(n-1)} \to \mathcal{Z}_n, (h, \breve{h})_\sharp P_{XY_{1:(n-1)}} = P_{Y_n} \otimes P_{Z_n}\}$. However, the pair $(h, \breve{h})$ yields an unintuitive structure $h(X, Y_{1:(n-1)}) \overset{d}{=} Y_n$, $\breve{h}(X, Y_{1:(n-1)}) \overset{d}{=} Z_n$. The following result shows that there is an equivalent pair that mirrors the decoder: $f(X, Y_{1:(n-1)}, \tilde{Y}_n^h) \overset{d}{=} Z_n$ where $\tilde{Y}_n^h := h(X, Y_{1:(n-1)})$ for a function $h : \mathcal{X} \times \mathcal{Y}_{1:(n-1)} \to \mathcal{Y}_n$.

Corollary 3.2. *Under the condition of Theorem 3.1, there holds*

$$W_p^p(P_{XY_{1:(n-1)}}, P_{GY_{1:(n-1)}}) = \inf_{(f,h)\in\mathcal{I}} \mathbb{E}_{P_{XY_{1:(n-1)}}} \Big\{ d^p(X, g \circ (\mathcal{A}_1 f) \circ (\mathcal{A}_0 h)(X, Y_{1:(n-1)})) \Big\}, \tag{8}$$

where $\mathcal{I} = \{(f, h) | f : \mathcal{X} \times \mathcal{Y}_{1:n} \to \mathcal{Z}_n, h : \mathcal{X} \times \mathcal{Y}_{1:(n-1)} \to \mathcal{Y}_n, ((\mathcal{A}_1 f) \circ (\mathcal{A}_0 h))_\sharp P_{XY_{1:(n-1)}} = (\bigotimes_{i=1}^n P_{Y_i}) \otimes P_{Z_n}\}$.

Remark 3.3. (i) Corollary 3.2 implies that if Y_n is missing, one can simply write the objective function by replacing Y_n with the $\tilde{Y}_n^h$, which predicts Y_n from $(X, Y_{1:(n-1)})$, in (6) of the complete data case. The functional constraints are derived in the same manner.

(ii) From the independence of $Y_{1:n}$, we can let $h(x, y_{1:(n-1)}) \equiv h(x)$. We leverage this fact in our experiments.

(iii) Let $S = Y_{1:(n-1)} \in \mathbb{R}^K, K = \sum_{i=1}^{n-1} k_i$, the set of all observable nuisance variables; $S = \emptyset$ if $n = 1$. The chain model reduces to the M2 model (Figure 1b) or the stacked model (Figure 1d), both are from Kingma et al. (2014), since we use a deterministic decoder.

So far, we have assumed that Y_n and $S = Y_{1:(n-1)}$ are independent. However, there are relevant cases in which Y and S are correlated, e.g., S is the age of a person and Y is the indicator of high income in socioeconomic datasets (cf. Louizos et al., 2016). In this setting, *conditional*, not joint, modeling is required and we need to derive the Wasserstein distance between $P_{XY|S}$ and $P_{GY|S}$ or between $P_{X|S}$ and $P_{G|S}$. We summarize these results in the Appendix B.

3.2. Building StrWAEs

Based on the previous theorem and corollary, we derive the objective function of WAEs for the three decoder models, depicted in Figures 1b to 1d, allowing for missing data. Remark 3.3 suggests that the chain model generally reduces to two cases: the case with only nuisance variables and the case with nuisance variables and labels. In the latter case, the goal is to find a representation that is independent of nuisance variables but has the information in the labels.

3.2.1. M2 Model

The M2 model has been considered for semi-supervised classification. It can be interpreted as the chain model in case $n = 1$. Here, variable Y represents the observed nuisance variation, and Z encodes the representation *invariant* to the unwanted variation in Y.

For the complete data, recall that the constraint defining $\mathcal{F}$ in (6) is equivalent to

$$f(X, Y) \overset{d}{=} Z, \quad Y \perp\!\!\!\perp f(X, Y). \tag{9}$$

If $\mathcal{D}$ is a divergence between the distributions of two random variables, such as the maximum mean discrepancy (MMD, Gretton et al., 2012; Rustamov, 2021) or Jensen-Shannon divergence (Goodfellow et al., 2014) and $\mathcal{H}$ is a criterion for measuring independence of two random variables, such as the HSIC or mutual information, then the constraints in (9) can be written as

$$\mathcal{D}(f_\sharp P_{XY} \| P_Z) = 0, \quad \mathcal{H}(Y, f(X, Y)) = 0.$$

Thus a penalized version of (6) is

$$\inf_f \{\mathbb{E}_{P_{XY}} d^p(X, g \circ (\mathcal{A}_1 f)(X, Y)) + \lambda_1 \mathcal{D}(f_\sharp P_{XY} \| P_Z) + \mu_1 \mathcal{H}(Y, f(X, Y))\} \tag{10}$$

for appropriate $\lambda_1, \mu_1 > 0$. The first term in (10) is the reconstruction error and the second term is the prior matching term, aiming to align the aggregate posterior with the prior distribution. The third term promotes independence between the latent variable Z and the label Y to achieve an invariant representation. In previous studies, this independence term was either arbitrarily included or implicit in the decoder architecture. However, in the WAE framework, it is *naturally* incorporated through the imposed constraint (9).

If Y is missing, the constraint set $\mathcal{I}$ in (8) is equivalent to $h(X) \overset{d}{=} Y,\ f(X, h(X)) \overset{d}{=} Z,\ h(X) \perp\!\!\!\perp f(X, h(X))$. Proceeding as above, we obtain

$$\begin{aligned}\inf_{f,h}\{&\mathbb{E}_{P_X}\, d^p(X, g \circ (\mathcal{A}_1 f)(X, h(X))) \\ &+ \lambda_1 \mathcal{D}((f \circ (\mathcal{A}_0 h))_\sharp P_X \| P_Z) \\ &+ \lambda_2 \mathcal{D}(h_\sharp P_X \| P_Y) + \mu_1 \mathcal{H}(h(X), f(X, h(X)))\}.\end{aligned} \tag{11}$$

for appropriate $\lambda_1, \lambda_2, \mu_1 > 0$. The catch here is that $\mathcal{D}(h_\sharp P_X \| P_Y)$ involves Y and thus cannot be estimated if the labels are completely missing in the sample. Fortunately, it can be estimated if they are *partially observed*. In this semi-supervised setting, P_Y can be estimated from the observed response variables, and the distribution $h_\sharp P_X$ of $h(X)$ can be estimated from the whole sample. So we need to merge (10) and (11), akin to how the loss function in a semi-supervised model involves a weighted sum of supervised and unsupervised losses (Yang et al., 2022). Note that (8) is valid even if the expectation is taken w.r.t. P_{XY}, i.e., as if Y is observed. If we define a random variable $\hat{Y}_h$ as $\hat{Y}_h = Y$ if Y is observed and $\hat{Y}_h = h(X)$ otherwise, then

$$\begin{aligned}&\mathbb{E}_{P_{X\hat{Y}_h}}\, d^p(X, g \circ (\mathcal{A}_1 f)(X, \hat{Y}_h)) + \lambda_1 \mathcal{D}(f_\sharp P_{X\hat{Y}_h} \| P_Z) \\ &\quad + \lambda_2 \mathcal{D}(P_{\hat{Y}_h} \| P_Y) + \mu_1 \mathcal{H}(\hat{Y}_h, f(X, \hat{Y}_h))\end{aligned} \tag{12}$$

is the desired combination of the reconstruction error and penalties that can be minimized by stochastic optimization.

Comparison with VAEs. In semi-supervised VAEs (Kingma et al., 2014), the variational posterior is *assumed* to be factorized as $q_\phi(y, z|x) = q_\phi(y|x) q_\phi(z|y, x)$ and is not genuine posterior induced by the Bayes theorem. The first and second factors correspond to $h(x)$ and $f(x, y)$ in $\mathcal{I}$ from Corollary 3.2. While the former factorization is rather arbitrary, the constraints in the encoder set $\mathcal{I}$ are derived from the first principle.

Connection to adversarial autoencoders. If $\mathcal{D}$ is chosen as the Jensen-Shannon divergence, $\mu = 0$, and $\hat{Y}_h \equiv h(X)$, then (12) recovers the reconstruction and regularization phases of the semi-supervised version of the adversarial autoencoder (AAE, Makhzani et al., 2016), provided that P_Y is replaced with an equiprobable multinomial distribution and optimization is carried out for each term. Since the classification phase of the AAE can be implemented by optimizing the third term $\mathcal{D}_{KL}(h_\sharp P_X \| P_Y)$ over the complete data, (12) provides a theoretical justification of the semi-supervised AAE; matching the distribution of $\hat{Y}_h$ directly to P_Y by minimizing the cited functional appears more natural than fitting to an equiprobable model. Note that both models possess an equivalent number of parameters; the AAE has the learning rates of the optimizer for each of the three phases, while the StrWAE has the penalty coefficients associated with its three penalties.[1]

3.2.2. Extended M2 model with possibly missing nuisance variables

The M2 model can be extended with two additional independent nuisance variables that can be missing. In Figure 1c, Y may represent a person's identity in her portrait, which may be missing, and S partially observed attributes in the VGGFace2 dataset (Cao et al., 2018). In this model, we assume that the number of identities may not be known a priori, which means the test data may contain portraits of new individuals that were not in the training data. In this setup we want to obtain a representation Z that is invariant to the two nuisance variables. Theorem 3.1 and 3.2 apply directly by replacing Y with (S, Y) and $\mathcal{Y}$ with $\mathcal{S} \times \mathcal{Y}$. We then minimize

$$\begin{aligned}&\mathbb{E}_{P_{X\hat{S}_{h_1}\hat{Y}_{h_2}}}\, d^p(X, g \circ (\mathcal{A}_1 f)(X, \hat{S}_{h_1}, \hat{Y}_{h_2})) \\ &\quad + \lambda_1 \mathcal{D}(f_\sharp P_{X\hat{S}_{h_1}\hat{Y}_{h_2}} \| P_Z) + \lambda_2 \mathcal{D}(P_{\hat{S}_{h_1}\hat{Y}_{h_2}} \| P_{SY}) \\ &\quad + \mu_1 \mathcal{H}((\hat{S}_{h_1}, \hat{Y}_{h_2}), f(X, \hat{S}_{h_1}, \hat{Y}_{h_2}))\end{aligned} \tag{13}$$

over f, g, h_1, h_2 where $\hat{S}_{h_1} = S$ if S is observed and $\hat{S}_{h_1} = h_1(X)$ otherwise, and $\hat{Y}_{h_2} = Y$ if Y is observed and $\hat{Y}_{h_2} = h_2(X)$ otherwise. We underscore that the number of identities may be *infinite*. Accordingly, it is necessary to embed Y into a Euclidean space.

3.2.3. Stacked generative model with possibly missing nuisance variables

The graphical model of Figure 1d is employed as the decoder in Louizos et al. (2016); Lopez et al. (2018) for learning invariant representations given nuisance variables. It can be posed as a chain model with $n = 2$. The response variable Y may represent the identity of a person, and the nuisance parameter S may represent the light condition of a picture X. In addition, it is required that Z_1 and S are independent in order to impose invariance of representation to the nuisance variable. That is, we want two different portraits of a person to have similar values of Z_1, and those of two different people to have quite distinct values of Z_1, even if the encoder does not know whose portraits they are,

[1] Makhzani et al. (2016) suggest to train AAE in several phases. This is equivalent to cyclically minimize the individual terms in (12); the number of penalty coefficients is equivalent to that of the learning rates for those phases.

and we also want Z_1 to represent something immune to the light condition S.

Note that the independence of Y and S enables us to carry out a *joint* modeling. As in Section 3.2.1 considering the additional independence constraint $Z_1 = g_2(\hat{Y}_h, f(X, S, \hat{Y}_h)) \perp\!\!\!\perp S$, the semi-supervised loss is derived as

$$\begin{aligned}&\mathbb{E}_{P_{XS\hat{Y}_h}} d^p(X, g \circ (\mathcal{A}_1 f)(X, S, \hat{Y}_h)) \\ &\quad + \lambda_1 \mathcal{D}(f_\sharp P_{XS\hat{Y}_h} \| P_Z) + \lambda_2 \mathcal{D}(P_{\hat{Y}_h} \| P_Y) \\ &\quad + \mu_1 \mathcal{H}(S, \hat{Y}_h, f(X, S, \hat{Y}_h)) \\ &\quad + \mu_2 \mathcal{H}(S, g_2(\hat{Y}_h, f(X, S, \hat{Y}_h)))\end{aligned} \tag{14}$$

for $\lambda_1, \lambda_2, \mu_1, \mu_2 > 0$, where the fourth term is aiming for the independence of the three terms, so one can utilize the dHSIC (Lopez et al., 2018).

Algorithm 1 describes the training procedure of StrWAEs. The hat notation denotes the empirical distribution or sample estimate.

Algorithm 1 Traning StrWAEs in Sections 3.2.1 to 3.2.3

Require: f: encoder, g: decoder, h: Y-embedder, Y: partially observed labels, S: observed labels (can be $\emptyset$), $\lambda_1, \lambda_2, \mu_1, \mu_2$: hyperparameters
Initialize f, g, h; $L_l \leftarrow 0, L_u \leftarrow 0$
repeat
  Sample $(x_l, s_l, y) \sim P_{XSY}, (x_u, s_u) \sim P_{XS}$
  Sample $z \sim P_Z$
  Compute $\begin{cases} G_l = g \circ (\mathcal{A}_1 f)(x_l, s_l, y), \\ G_u = g \circ (\mathcal{A}_1 f) \circ (\mathcal{A}_0 h)(x_u, s_u) \end{cases}$
  $L_l \leftarrow d^p(X_l, G_l) + \lambda_1 \hat{\mathcal{D}}(f_\sharp \hat{P}_{XSY} \| \hat{P}_Z)$
    $+ \lambda_2 \hat{\mathcal{D}}(h_\sharp \hat{P}^l_X \| \hat{P}_Y) + \mu_1 \hat{\mathcal{H}}(s_l, y, f(x_l, s_l, y))$
    $+ \mu_2 \hat{\mathcal{H}}(s_l, g_2(y, f(x_l, s_l, y)))$
  $L_u \leftarrow d^p(X_u, G_u) + \lambda_1 \hat{\mathcal{D}}((f \circ (\mathcal{A}_0 h))_\sharp \hat{P}_{XS} \| \hat{P}_Z)$
    $+ \lambda_2 \hat{\mathcal{D}}(h_\sharp \hat{P}^u_X \| \hat{P}_Y)$
    $+ \mu_1 \hat{\mathcal{H}}(s_u, h(x_u), f(x_u, s_u, h(x_u)))$
    $+ \mu_2 \hat{\mathcal{H}}(s_u, g_2(h(x_u), f(x_u, s_u, h(x_u))))$
  Take a gradient descent step [a] on $L_l + L_u$
until f, g, h converge

[a] Each term can be optimized separately; see footnote in §3.2.1.

4. Experiments

We experimented StrWAEs with various real-world datasets.[2] The generative model in Section 3.2.1 is applied for semi-supervised learning and conditional generation on the MNIST and SVHN (Netzer et al., 2011) datasets. The models in Sections 3.2.2 and 3.2.3 are used for learning conditional generation on the VGGFace2 datasets (Cao et al., 2018) and invariant representation on the Extended Yale B dataset (Georghiades et al., 2001; Lee et al., 2005). We compared StrWAEs with semi-supervised VAE models that are capable of conditional generation. The metric d was chosen to be the Euclidean distance and $p = 2$ was used for the power, so that the reconstruction error equals the mean squared error. For the independence criterion $\mathcal{H}$, dHSIC was employed as in Lopez et al. (2018). For the distributional divergence $\mathcal{D}$, we used the Jensen-Shannon divergence (implemented with GAN) to match the encoded and prior distributions. In light of Remark 2.2, we used Leaky ReLU activations for the decoder to assure injectivity.

4.1. Semi-supervised Learning

We performed semi-supervised classification tasks on the MNIST dataset with 100 labeled observations and the SVHN dataset with 1000 labeled observations. All digits were sampled evenly. The class label Y was embedded in the 10-dimensional probability simplex. The latent distribution P_Z was chosen as a standard normal. The divergence $\mathcal{D}(P_{\hat{Y}_h} \| P_Y)$ in (12) was chosen as the sum of the cross entropy (if Y is observed, 0 otherwise) and the MMD between two random variables. To train this term using mini-batch gradient descent, labeled observations in a batch were sampled with maintaining the proportion in the overall observed labels. Note that this model was trained end-to-end. The classification accuracy of the trained StrWAE was compared with the Gaussian mixture VAE (GMVAE, Dilokthanakul et al., 2016), semi-supervised VAE (M1+M2, Kingma et al., 2014), AAE (Makhzani et al., 2016), CCVAE (Joy et al., 2021), and the M2 model with the Smooth-ELBO objective (Feng et al., 2021). In addition, conditional generation was conducted by sampling a latent variable Z from P_Z and setting Y as the one-hot encoding of the digit from the test dataset. Then the image was generated by computing the decoder output $g(Y, Z)$. To assess the quality of this conditional generation procedure, the classification accuracy of the generated images on a separate, pretrained digit classifier were computed. The pretrained classifiers had 99.32% (MNIST) and 96.46% (SVHN) of accuracy. To further assess the ability of class-preserving generation, we also examined the accuracy of generated images with the embedding $\hat{Y} = h(X)$ as input to the decoder instead of Y.

The result is summarized in Table 1. The StrWAE achieved the highest classification accuracy on the MNIST dataset, and the second-highest accuracy on the SVHN dataset. Furthermore, StrWAE showed a substantially better performance in conditional generation, surpassing other models by a large margin and generation performance did not differ significantly whether an actual label or a predicted label

[2] Code available at https://github.com/comp-stat/StrWAE

Table 1: Semi-supervised classification and generation performance. The classification accuracy for the M1+M2 model is excerpted from Kingma et al. (2014), as the reported level of accuracy was not reproducible. † In the SVHN dataset, clusters were generated based on background rather than the labels, resulting in low accuracies.

Model	MNIST(100)			SVHN(1000)		
	Classification	Conditional Generation		Classification	Conditional Generation	
		Y	$\hat{Y}$		Y	$\hat{Y}$
M1+M2	96.67	-	-	63.98	-	-
GMVAE†	86.40(±1.80)	87.44 (±1.45)	83.00 (±2.90)	17.45 (±2.20)	13.57 (±1.19)	17.54 (±0.49)
AAE	98.10 (±0.35)	36.69 (±1.74)	35.75 (±2.10)	82.97 (±1.25)	82.99 (±3.17)	69.65 (±3.54)
CCVAE	93.18 (±0.45)	16.03 (±0.57)	67.42 (±5.71)	**90.02 (±0.07)**	14.92 (±0.89)	17.55 (±1.89)
Smooth-ELBO	95.97 (±1.95)	27.27 (±1.42)	11.75 (±1.34)	84.51 (±0.11)	81.85 (±0.09)	52.99 (±1.98)
StrWAE	**98.69 (±0.04)**	**96.06 (±0.57)**	**94.74 (±0.55)**	85.59 (±0.22)	**98.80 (±0.44)**	**82.69 (±0.32)**

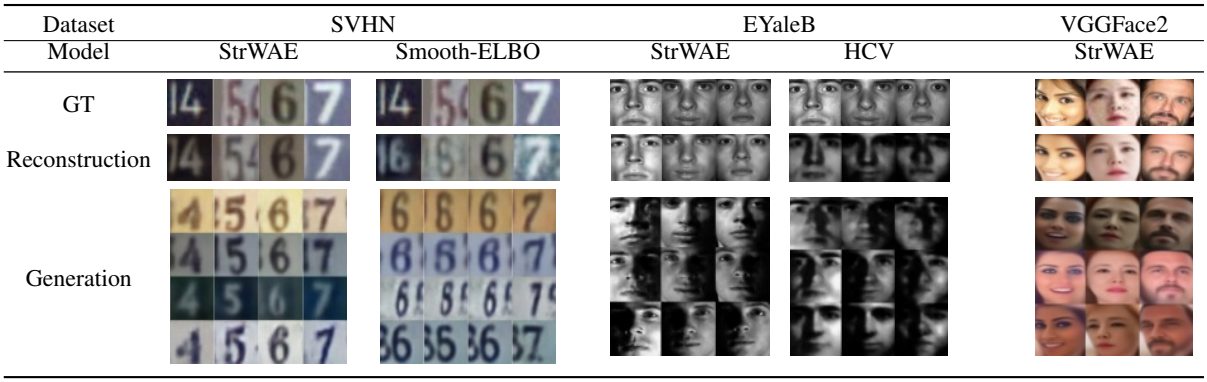


Figure 2: Class-preserving generation examples

were given as input. Compared to AAE, all the terms in (12) use the full data, and additionally, an independence penalty appears naturally. These differences are attributed to the relatively good conditional generation and class-preserving generation performance of StrWAE. We hypothesize that CCVAE has the highest classification accuracy on the SVHN dataset because CCVAE is intended for multi-label settings rather than assigning a single label to an input, and the images in the SVHN dataset contain extra digits in the background in addition to the labels. The generation quality of Smooth-ELBO was poor while its classification accuracy was high. It is likely due to that the gap between the likelihood and the smooth-ELBO still exists even after the smooth-ELBO incorporates the ELBO and the classification loss. StrWAE does not suffer this kind of difficulties as it directly targets the Wasserstein distance between the source and target distributions. In the MNIST dataset, latent vectors of GMVAE clustered on different digits. Consequently GMVAE outperformed the other semi-supervised VAE models in conditional generation. However, this was not the case for predicting digits. This suggests that in the VAE framework, clustering latent vectors is desirable for conditional generation and semi-supervised learning is better for predicting missing labels. On the contrary, the WAE framework performed well on both tasks.

Figure 2 provides a qualitative comparison of class-preserving generation performance on the SVHN dataset. The second row was generated by putting the encoder output as input to the decoder. The last four rows contains the class-preserving generation results. Images generated by the StrWAE all retained the digit class of the input, a phenomenon not observed in Smooth-ELBO. Figure 3 shows the style transfer results, where the class label was predicted by $h(X)$ from the source image X and from the target image X' its "style of handwriting" (representation invariant to digit variation) $f(X', h(X'))$ was estimated. The output was generated by computing $g(h(X), f(X', h(X')))$. Images generated from the StrWAE successfully inherited information from different target and source images. For more results, see Appendix F.

4.2. Conditional Generation Using Embedded Variables

Image data. We further investigated the conditional generation capability of StrWAEs using the VGGFace2 dataset (Cao et al., 2018). This dataset comprises 3.14 million training images, featuring faces from a total of 8,631 subjects, and 169,000 test images from 500 subjects. Binary attributes such as gender, presence of sunglasses, and mouth state (open or closed) are partially observed and documented for a subset of 30,000 images. Here, we let Y be the identity

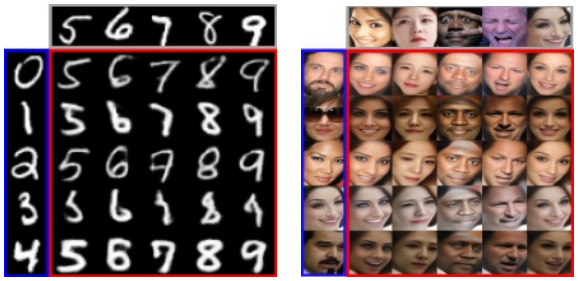


Figure 3: Style transfer examples of StrWAEs. Left most column: source style. Top most row: target class.

of the person in the image and S be the vector of attributes. The generative model for this dataset is the same as Section 3.2.2. In order to embed Y and S to the Euclidean space $\mathbb{R}^B$ where B does not necessarily depend on the number of categories, we employed the entity embedding network (Guo & Berkhahn, 2016) for the observed labels. The trained embedding network naturally becomes pretrained encoders h_1 and h_2 in (13), which are held fixed when training the rest of the network. A by-product of this embedding is that it is even possible to impute a person's identity not present in training data.

The class-preserving generation and style transfer tasks were conducted in the same manner as MNIST. In Figure 2, although images of persons who were *not* in the training data were used, the StrWAE was able to successfully generate images retaining the identity with varying identity-invariant features, e.g., camera angle, lighting condition. In the style transfer task (Figure 3), the generated images possess the styles from the source data and tend to preserve the specified attribute of the target data. For example, the generated images tend to have open mouth if the target image has mouth wide open. In addition, we also tried generating samples with manipulated attributes. Since the "attribute encoder" h_2 embeds S in the Euclidean space, we could extrapolate input S to decoder g beyond 0 and 1. For this attribute manipulation task, we compared results with Fader Networks (Lample et al., 2018) trained with a similar architecture. In the attribute manipulation task, we could successfully generate images with the desired attributes changed. In Figure 4, letting the "Beard" attribute positive produced images having shaggy beard; making it negative produced images without beard. For the Fader Networks, we extrapolated the attribute scores to a large magnitude as far as ± 400, but it only distorted the original image. When we manipulated the "beard" attribute, we observed that both the mouth state (open or closed) and gender attributes were affected alongside the beard attribute in Fader Networks. This suggests that the latter model was not successful in obtaining representations invariant to nuisance attributes. On the other hand, StrWAE was relatively immune to the nuisance information.

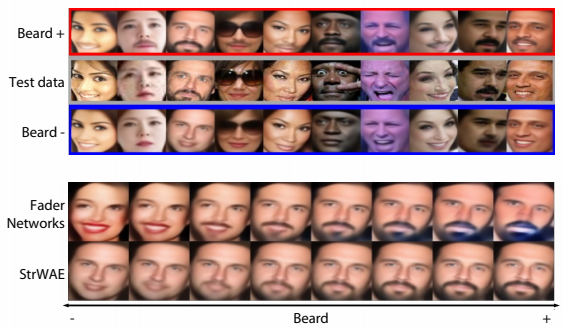


Figure 4: Attribute manipulation and interpolation examples on the VGGFace2 dataset.

4.3. Invariant representations

The same structure as Section 3.2.3 was used to test the ability of StrWAEs to learn invariant representations of controlled photographs, where the control variable is the light direction. The cropped Extended Yale B dataset (Georghiades et al., 2001; Lee et al., 2005) comprises of facial images of 38 human subjects in various lighting conditions. For each subject, the pictures of the person are split into training and test data with a fixed ratio, resulting in 1,664 and 750 images for the training and test respectively. We set the identity of the image as Y and the lighting condition (elevation and azimuth of the light direction normalized into $[-1, 1] \times [-1, 1]$) as S. In the training stage, we first pretrained the h in (14) to estimate Y and fixed h for the rest of the training procedure. In consequence, we were able to encode and decode the test data without the information about Y by replacing it with $h(X)$.

Table 2 and Figure 5 collect the results for predicting identity and lighting direction (grouped in five and continuous), and image generation, following Lopez et al. (2018). Recall that the goal is to learn latent representations $Z_1 = g_2(h(X), f(X, S, h(X)))$ that are invariant to the light direction while retaining the identity information. The Z_1 encoded by StrWAE shows better performance in predicting Y and worse in predicting S on the test dataset than others, suggesting better invariant representation; training without the HSIC penalty led to performance degradation. The t-SNE visualization of Z_1 (Figure 5) accords with this result, showing noticeable separation with respect to person (Y), but not with respect to lighting (S). Sample images were generated by using the representation of the test images but setting $S = (\pm 0.3, \pm 0.3)$; see Figure 2. StrWAE produced reconstructions closer to the input than HCV and perturbing S only kept the identity of the input in the generated images. The sharpness (Tolstikhin et al., 2018) and the Fréchet inception distance (FID) scores (Heusel et al., 2017) show that StrWAE produced sharper images than HCV, confirming the visual inspection. Since the sample generation was conducted by varying the variable S, the

Table 2: Invariant representation of Extended Yale B. RF=random forest, Logistic=logistic regression, Linear=linear regression. Classification accuracy for discrete and mean squared error for continuous variables.

Model	ID accuracy	Lighting group (Acc.)		Lighting direction (MSE)		Generation	
		RF	Logistic	RF	Linear	FID	Sharpness
VAE	0.71	0.73	0.74	0.03	0.07	222	7.67e-5
HCV	0.72	0.51	0.46	0.18	0.21	232	8.27e-5
StrWAE	**0.98**	**0.23**	**0.24**	**0.25**	**0.24**	**55.7**	**3.94e-4**

generated samples should be different from the test data with scarce images and thus the FID scores should be taken with caution, though. Overall, that StrWAE uses deterministic encoder and decoder unlike the VAE-based model likely contributed to achieving the desired representation and good conditional generation performance at the same time.

Speech Data. To investigate the versatility of StrWAEs extends beyond facial images, we conducted semi-supervised classification and conditional generation experiments on the Mini Speech Recognition dataset[3] comprising eight short words, which were used as labels. Each word consists of 1,000 pronunciations, and we employed 9:1 train-test split on the dataset. Notably, we trained our model with only 20% of the data containing labels. All audio samples were transformed into mel-spectrograms. The inclusion of speech recognition tasks within VAE frameworks is limited by the inherent complexity of audio data. Our experimental results with AAE highlight this challenge. While the latent variables in AAE fail to capture crucial audio information, StrWAE demonstrates its capability to effectively encode intricate audio data and facilitate conditional generation.

Table 3: Speech recognition results.

Model	Classification	Conditional Generation	
		Y	$\hat{Y}$
StrWAE	**60.88**	**57.63**	**37.50**
AAE	24.03	13.9	13.9

Tabular data. We also explored StrWAEs' possibility in fair representation learning using tabular data. Due to the lack of space, the details are provided in Appendix G.

We conclude this section by emphasizing that these various flavors of invariant representation learning are possible within the single framework of WAEs.

5. Discussion

We have shown that the WAE framework is rich enough to handle various conditional independence structures, leading to solid formulations of invariant learning problems, in the setting that the decoder is structured and nuisance information is (partially) available. Coining penalties to promote the assumed factorization of the variational posterior (encoder), often mimicking the factorization, or conditional independence structure, of the decoder, is a commonly found approach in the vast literature on VAEs, whether nuisance information is present (Kingma et al., 2014; Louizos et al., 2016; Lopez et al., 2018; Liu et al., 2022) or not (Higgins et al., 2017; Kim & Mnih, 2018; Chen et al., 2018). The sheer number of proposed penalties indicates the difficulty and lack of principles in determining the encoder structure matching that of the decoder. The WAE framework discussed in this paper can overcome these pitfalls.

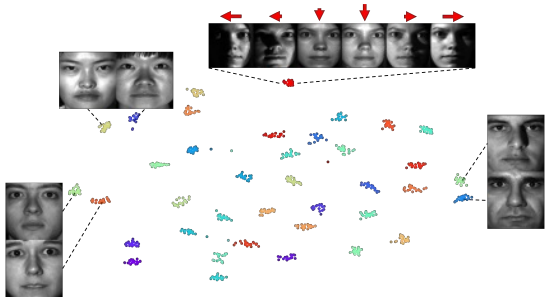


Figure 5: t-SNE map of Z_1 colored by identities in Extended Yale B. Each person formed a cluster within which various lighting conditions were collected. Similarly-looking faces appear to locate closely.

To this end, we provide in Appendix C some ablation studies and provide a guide on selecting the penalty parameters for StrWAEs.

Since the penalty parameters are finite in practice, the choice of the divergence $\mathcal{D}$ and independence criterion $\mathcal{H}$ may affect the performance of StrWAEs. In Appendix D, we compare popular basic divergences. It appears that those that require adversarial training perform better.

The WAE literature has focused on improving the divergence that matches the prior P_Z and the aggregated posterior Q_Z in (2), e.g., sliced Wasserstein distance (Kolouri et al., 2019), Sinkhorn divergence (Patrini et al., 2020; Genevay et al., 2018), and relational divergences (Xu et al., 2020; Nguyen et al., 2021). Note that these exotic divergences are also compatible with our framework. Investigating which divergences will perform effectively in the structured settings would be an interesting direction.

[3] https://www.tensorflow.org/tutorials/audio/simple_audio

Acknowledgements

This work was supported by the AI-Bio Research Grant through Seoul National University, by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the South Korean government (MSIT) [No. 2021-0-01343-004, Artificial Intelligence Graduate School Program (Seoul National University)], and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00337691).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Cao, Q., Shen, L., Xie, W., Parkhi, O. M., and Zisserman, A. VGGFace2: A dataset for recognising faces across pose and age. In *Proc. 13th IEEE Int. Conf. Automatic Face & Gesture Recognition (FG 2018)*, pp. 67–74. IEEE, 2018.

Chen, R. T., Li, X., Grosse, R. B., and Duvenaud, D. K. Isolating sources of disentanglement in variational autoencoders. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2018)*, volume 31, 2018.

Dilokthanakul, N., Mediano, P. A., Garnelo, M., Lee, M. C., Salimbeni, H., Arulkumaran, K., and Shanahan, M. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

Feng, H.-Z., Kong, K., Chen, M., Zhang, T., Zhu, M., and Chen, W. SHOT-VAE: Semi-supervised Deep Generative Models with Label-aware ELBO approximations. *Proc. AAAI Conf. Artif. Intell. (AAAI 2021)*, 35(8):7413–7421, 2021.

Genevay, A., Peyre, G., and Cuturi, M. Learning Generative Models with Sinkhorn Divergences. In *Proc. 21st Int. Conf. Artif. Intell. Statist. (AISTATS 2018)*, volume 84, pp. 1608–1617. PMLR, 2018.

Georghiades, A. S., Belhumeur, P. N., and Kriegman, D. J. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(6):643–660, 2001.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2014)*, volume 27, 2014.

Gretton, A., Fukumizu, K., Teo, C., Song, L., Schölkopf, B., and Smola, A. A kernel statistical test of independence. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2007)*, volume 20, 2007.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *J. Mach. Learn. Res.*, 13(25):723–773, 2012.

Guo, C. and Berkhahn, F. Entity Embeddings of Categorical Variables. *arXiv preprint arXiv:1604.06737*, 2016.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2017)*, volume 30, 2017.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *Proc. Int. Conf. Learn. Represent. (ICLR 2017)*, 2017.

Joy, T., Schmon, S., Torr, P., Narayanaswamy, S., and Rainforth, T. Capturing label characteristics in VAEs. In *Proc. Int. Conf. Learn. Represent (ICLR 2021)*, 2021.

Kechris, A. *Classical descriptive set theory*, volume 156. Springer Sci. & Bus. Media, 2012.

Kim, H. and Mnih, A. Disentangling by factorising. In *Proc. Int. Conf. Mach. Learn. (ICML 2018)*, volume 80, pp. 2649–2658. PMLR, 2018.

Kim, Y.-g., Lee, K., and Paik, M. C. Conditional Wasserstein Generator. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(06):7208–7219, 2023.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *Proc. Int. Conf. Learn. Represent. (ICLR 2014)*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *Proc. Int. Conf. Learn. Represent. (ICLR 2014)*, 2014.

Kingma, D. P., Rezende, D. J., Mohamed, S., and Welling, M. Semi-Supervised Learning with Deep Generative Models. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2014)*, volume 27, 2014.

Kolouri, S., Pope, P. E., Martin, C. E., and Rohde, G. K. Sliced-Wasserstein Auto-Encoders. In *Proc. Int. Conf. Learn. Represent. (ICLR 2019)*, 2019.

Lample, G., Zeghidour, N., Usunier, N., Bordes, A., Denoyer, L., and Ranzato, M. Fader Networks: Manipulating Images by Sliding Attributes. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2018)*, volume 31, 2018.

Lee, K.-C., Ho, J., and Kriegman, D. J. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(5):684–698, 2005.

Liu, J., Li, Z., Yao, Y., Xu, F., Ma, X., Xu, M., and Tong, H. Fair representation learning: An alternative to mutual information. In *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining (KDD 2022)*, pp. 1088–1097, 2022.

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. On the variance of the adaptive learning rate and beyond. In *Proc. Int. Conf. Learn. Represent. (ICLR 2020)*, 2020.

Lopez, R., Regier, J., Jordan, M. I., and Yosef, N. Information constraints on auto-encoding variational Bayes. In *Adv. Neural Inf. Process. Syst. (NeurIPS 2018)*, volume 31, 2018.

Louizos, C., Swersky, K., Li, Y., Welling, M., and Zemel, R. S. The variational fair autoencoder. In *Proc. Int. Conf. Learn. Represent. (ICLR 2016)*, 2016.

Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. Adversarial autoencoders. In *Proc. Int. Conf. Learn. Represent. (ICLR 2016)*, 2016.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Nguyen, K., Nguyen, S., Ho, N., Pham, T., and Bui, H. Improving Relational Regularized Autoencoders with Spherical Sliced Fused Gromov Wasserstein. In *Proc. Int. Conf. Learn. Represent. (ICLR 2021)*, 2021.

Patrini, G., van den Berg, R., Forre, P., Carioni, M., Bhargav, S., Welling, M., Genewein, T., and Nielsen, F. Sinkhorn autoencoders. In *Proc. Uncertain. Artif. Intell. (UAI 2020)*, volume 115, pp. 733–743. PMLR, 2020.

Pratelli, A. On the equality between Monge's infimum and Kantorovich's minimum in optimal mass transportation. *Annales de l'I.H.P. Probabilités et statistiques*, 43(1):1–13, 2007.

Puthawala, M., Kothari, K., Lassas, M., Dokmanić, I., and De Hoop, M. Globally injective relu networks. *J. Mach. Learn. Res.*, 23(1):4544–4598, 2022.

Rustamov, R. M. Closed-form expressions for maximum mean discrepancy with applications to Wasserstein auto-encoders. *Stat*, 10:e329, 2021.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schölkopf, B. Wasserstein auto-encoders. In *Proc. Int. Conf. Learn. Represent. (ICLR 2018)*, 2018.

Tschannen, M., Bachem, O., and Lucic, M. Recent advances in autoencoder-based representation learning. In *Third workshop on Bayesian Deep Learning (NeurIPS 2018)*, 2018.

Van Engelen, J. E. and Hoos, H. H. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440, 2020.

Xu, H., Luo, D., Henao, R., Shah, S., and Carin, L. Learning Autoencoders with Relational Regularization. In *Proc. Int. Conf. Mach. Learn. (ICML 2020)*, volume 119, pp. 10576–10586. PMLR, 2020.

Yang, X., Song, Z., King, I., and Xu, Z. A survey on deep semi-supervised learning. *IEEE Trans. Knowl. Data Eng.*, 35(9):8934–8954, 2022.

A. Proofs

A.1. Proof of Theorem 2.1

Proof. Under the conditions of the theorem statement, the Monge-Kantorovich equivalence holds (Pratelli, 2007, Theorem B):

$$W_p^p(P_X, P_G) = \inf_{T:\mathcal{X}\to\mathcal{X}:T_\sharp P_X = P_G} \mathbb{E}_{P_X}\, d^p(X, T(X)).$$

Hence it suffices to show that

$$\inf_{f:\mathcal{X}\to\mathcal{Z}:f_\sharp P_X = P_Z} \int_{\mathcal{X}} d^p(x, g(f(x)))dP_X(x) = \inf_{T:\mathcal{X}\to\mathcal{X}:T_\sharp P_X = P_G} \int_{\mathcal{X}} d^p(x, T(x))dP_X(x).$$

First observe that, for any measurable $f : \mathcal{X} \to \mathcal{Z}$ such that $f_\sharp P_X = P_X f^{-1} = P_Z$, we have $g \circ f : \mathcal{X} \to \mathcal{X}$ measurable and for any Borel set $E \subset \mathcal{X}$

$$\begin{aligned}(g \circ f)_\sharp P_X(E) = P_X(g \circ f)^{-1}(E) &= P_X(f^{-1}(g^{-1}(E))) \\ &= g_\sharp[P_X f^{-1}](E) = g_\sharp P_Z(E) = P_G(E)\end{aligned}$$

or $(g \circ f)_\sharp P_X = P_G$, resulting in

$$\inf_{f:\mathcal{X}\to\mathcal{Z}:f_\sharp P_X = P_Z} \int_{\mathcal{X}} d^p(x, g(f(x)))dP_X(x) \geq \inf_{T:\mathcal{X}\to\mathcal{X}:T_\sharp P_X = P_G} \int_{\mathcal{X}} d^p(x, T(x))dP_X(x).$$

For the opposite direction, suppose $T : \mathcal{X} \to \mathcal{X}$ that is measurable and satisfies $T_\sharp P_X = P_G = g_\sharp P_Z$ approximately attains the infimum, i.e., given $\epsilon > 0$,

$$\int_{\mathcal{X}} d^p(x, T(x))dP_X(x) < \inf_{T':\mathcal{X}\to\mathcal{X}:T'_\sharp P_X = P_G} \int_{\mathcal{X}} d^p(x, T'(x))dP_X(x) + \epsilon.$$

Consider any Borel set $B \subset \mathcal{Z}$ such that B^c is a null set of P_Z. From the Lusin-Suslin theorem (Corollary 15.2 of Kechris, 2012), $g(B)$ is Borel measurable because g is an injective Borel measurable function. Then $T_\sharp P_X = g_\sharp P_Z$ entails that

$$1 = P_Z(B) = P_Z(g^{-1}(g(B))) = g_\sharp P_Z(g(B)) = T_\sharp P_X(g(B)) = P_X(T^{-1}(g(B))). \tag{15}$$

Since g is injective, for each $x \in g(\mathcal{Z})$ there is a unique z such that $x = g(z)$. Let us denote this z by $g^{-1}(x)$. Pick any $z_0 \in \mathcal{Z}$ and define $g^\dagger : \mathcal{X} \to \mathcal{Z}$ as

$$g^\dagger(x) = \begin{cases} g^{-1}(x), & x \in g(\mathcal{Z}) \\ z_0, & \text{otherwise.} \end{cases}$$

If we let $f = g^\dagger \circ T$, then for each $x \in T^{-1}(g(B))$,

$$g \circ f(x) = g \circ g^\dagger \circ T(x) = g(g^\dagger(T(x))) = g(g^{-1}(T(x)) = T(x)$$

since $T(x) \in g(B) \subset g(\mathcal{Z})$. From (15), $T = g \circ f$, P_X-a.e. Notice that $g^\dagger$ is a left inverse of g and it is measurable again by Corollary 15.2 of Kechris (2012). Thus for any Borel set $F \subset \mathcal{Z}$,

$$\begin{aligned} f_\sharp P_X(F) = P_X(g^\dagger \circ T)^{-1}(F) &= P_X(T^{-1}((g^\dagger)^{-1}(F))) \\ &= T_\sharp P_X((g^\dagger)^{-1}(F)) \\ &= P_G((g^\dagger)^{-1}(F)) \\ &= g_\sharp P_Z((g^\dagger)^{-1}(F)) \\ &= P_Z(g^{-1}((g^\dagger)^{-1}(F))) \\ &= P_Z((g^\dagger \circ g)^{-1}(F)) = P_Z(F), \end{aligned}$$

which shows $f_\sharp P_X = P_Z$. Therefore,

$$\begin{aligned}\inf_{f':\mathcal{X}\to\mathcal{Z}:f'_\sharp P_X=P_Z}\int_{\mathcal{X}} d^p(x,g(f'(x)))dP_X(x) &\le \int_{\mathcal{X}} d^p(x,g(f(x)))dP_X(x)\\ &= \int_{\mathcal{X}} d^p(x,T(x))dP_X(x)\\ &< \inf_{T':\mathcal{X}\to\mathcal{X}:T'_\sharp P_X=P_G}\int_{\mathcal{X}} d^p(x,T'(x))dP_X(x)+\epsilon.\end{aligned}$$

Since $\epsilon > 0$ is arbitrary, we are done. □

A.2. Proof of Theorem 3.1

In order to prove the claim, we need the following result. A similar result can be found in (Kim et al., 2023, Theorem 2).

Proposition A.1. *Let random variables (X, X', Y) on $\mathcal{X}\times\mathcal{X}\times\mathcal{Y}$ satisfies $(X,Y)\sim P_{XY}$, $(X',Y)\sim P_{X'Y}$, and $Y\sim P_Y$. In particular the Y-marginals of P_{XY} and $P_{X'Y}$ are both P_Y. Then,*

$$W_p^p(P_{XY},P_{X'Y}) = \mathbb{E}_{P_Y} W_p^p(P_{X|Y},P_{X'|Y}).$$

Proof. Recall that

$$W_p^p(P_{XY},P_{X'Y}) = \inf_{\gamma_{XYX'Y'}\in\Pi(P_{XY},P_{X'Y})}\int_{\mathcal{X}\times\mathcal{Y}\times\mathcal{X}\times\mathcal{Y}} \tilde{d}^p((x,y),(x',y'))d\gamma_{XYX'Y'}(x,y,x',y')$$

where $\Pi(P_{XY},P_{X'Y})$ is the set of joint distributions of (X,Y,X',Y') with marginals P_{XY} and $P_{X'Y}$.

For any $\gamma_{XYX'Y'}\in\Pi(P_{XY},P_{X;Y})$,

$$\begin{aligned}&\int_{\mathcal{X}\times\mathcal{Y}\times\mathcal{X}\times\mathcal{Y}} \tilde{d}^p((x,y),(x',y'))d\gamma_{XYX'Y'}(x,y,x',y')\\ &\ge \int_{\mathcal{X}\times\mathcal{Y}\times\mathcal{X}\times\mathcal{Y}} d^p(x,x')d\gamma_{XYX'Y'}(x,y,x',y')\\ &\ge \int_{\mathcal{X}\times\mathcal{Y}\times\mathcal{X}\times\mathcal{Y}} d^p(x,x')I_{\{y=y'\}}d\gamma_{XYX'Y'}(x,y,x',y')\\ &= \int_{\mathcal{Y}\times\mathcal{Y}}\left[\int_{\mathcal{X}\times\mathcal{X}} d^p(x,x')d\gamma_{XX'|YY'}(x,x'|y,y')\right] I_{\{y=y'\}}d\gamma_{YY'}(y,y')\\ &= \int_{\mathcal{Y}}\left[\int_{\mathcal{X}\times\mathcal{X}} d^p(x,x')d\gamma_{XX'|YY'}(x,x'|y,y)\right] dP_Y(y)\\ &\ge \int_{\mathcal{Y}} W_p^p(P_{X|y},P_{X'|y})dP_Y(y).\end{aligned}$$

The last line holds because $\gamma_{XX'|YY'}(\cdot|y,y')\delta_y\in\Pi(P_{X|y},P_{X'|y})$. Therefore,

$$W_p^p(P_{XY},P_{X'Y}) \ge \mathbb{E}_{P_Y} W_p^p(P_{X|Y},P_{X'|Y}).$$

For the opposite direction, fix $\epsilon>0$ and let $\gamma_{XX'|y}\in\Pi(P_{X|y},P_{X'|y})$ approximately attains $W_p^p(P_{X|y},P_{X'|y})$, i.e.,

$$\int_{\mathcal{X}\times\mathcal{X}} d^p(x,x')d\gamma_{XX'|y}(x,x') < W_p^p(P_{X|y},P_{X'|y})+\epsilon.$$

Taking expectation with respect to P_Y, we obtain

$$\begin{aligned}\mathbb{E}_{P_Y} W_p^p(P_{X|y},P_{X'|y})+\epsilon &> \int_{\mathcal{Y}}\int_{\mathcal{X}\times\mathcal{X}} d^p(x,x')d\gamma_{XX'|y}(x,x')dP_Y(y)\\ &= \int_{\mathcal{X}\times\mathcal{Y}\times\mathcal{X}\times\mathcal{Y}} \tilde{d}^p((x,y),(x',y'))\delta_y(y')d\gamma_{XX'|y}(x,x')d\gamma_{YY'}(y,y')\\ &\ge W_p^p(P_{XY},P_{X'Y})\end{aligned}$$

where $\gamma_{YY'}$ is a joint distribution whose marginals are both P_Y. The last line holds since the measure π such that $\pi(B) = \int_B \delta_y(y') d\gamma_{XX'|y}(x, x') d\gamma_{YY'}(y, y')$ for any Borel set $B \in \mathcal{X} \times \mathcal{Y} \times \mathcal{X} \times \mathcal{Y}$ satisfies $\pi \in \Pi(P_{XY}, P_{X'Y})$. Sending $\epsilon \to 0$,

$$W_p^p(P_{XY}, P_{X'Y}) \leq \mathbb{E}_{P_Y} W_p^p(P_{X|Y}, P_{X'|Y}).$$

□

Proof of Theorem 3.1. It is sufficient to prove that only if $n = 1$. The case of $n \geq 2$ can be proven similarly by substituting $Y_{1:n}, P_{Y_1} \otimes \cdots \otimes P_{Y_n}$ for Y, P_Y respectively. Let $A \subset \mathcal{Y}$ be such that $P_Y(A) = 1$. Fix $y \in A$. From Theorem 2.1,

$$W_p^p(P_{X|y}, P_{G|y}) = \inf_{f_y : \mathcal{X} \to \mathcal{Z}, f_{y\sharp} P_X = P_Z} \mathbb{E}_{P_{X|y}} d^p(X, g_y(f_y(X))).$$

Thus for any f_y satisfying the constraint,

$$W_p^p(P_{X|y}, P_{G|y}) \leq \mathbb{E}_{P_{X|y}} d^p(X, g_y(f_y(X)))$$

Define $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ as $(x, y) \mapsto f_y(x)$. Then the above inequality holds P_Y-a.e. and $f \in \mathcal{F}$, where

$$\mathcal{F} = \{f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z} \mid f(\cdot, y)_\sharp P_{X|y} = P_Z,\ P_Y\text{-a.e.},\ f \text{ is measurable}\}.$$

Recall that $g_y = g(y, \cdot)$. Take expectation with respect to P_Y on both sides and apply Proposition A.1 to have

$$W_p^p(P_{XY}, P_{GY}) \leq \mathbb{E}_{P_{XY}} d^p(X, g(Y, f(X, Y))) \tag{16}$$

for any $f \in \mathcal{F}$.

Given $\epsilon > 0$, let $f_y^* : \mathcal{X} \to \mathcal{Z}$ approximately attains $W_p^p(P_{X|y}, P_{G|y})$, i.e., $f_{y\sharp}^* P_X = P_Z$ and

$$\mathbb{E}_{P_{X|y}} d^p(X, g_y(f_y^*(X))) < W_p^p(P_{X|y}, P_{G|y}) + \epsilon.$$

Define $f^* : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, $(x, y) \mapsto f_y^*(x)$. Taking expectation with respect to P_Y on both sides and applying Proposition A.1,

$$\mathbb{E}_{P_{XY}} d^p(X, g(Y, f^*(X, Y))) < W_p^p(P_{XY}, P_{GY}) + \epsilon.$$

Since $f^* \in \mathcal{F}$,

$$\inf_{f \in \mathcal{F}} \mathbb{E}_{P_{XY}} d^p(X, g(Y, f(X, Y))) \leq \mathbb{E}_{P_{XY}} d^p(X, g(Y, f^*(X, Y))) < W_p^p(P_{XY}, P_{GY}) + \epsilon.$$

Sending $\epsilon \to 0$ yields

$$\inf_{f \in \mathcal{F}} \mathbb{E}_{P_{XY}} d^p(X, g(Y, f(X, Y))) \leq W_p^p(P_{XY}, P_{GY}). \tag{17}$$

Combining (16) and (17) yields the desired equality.

It remains to show that the $\mathcal{F}$ is equal to

$$\{f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z} \text{ measurable} \mid \tilde{f}_\sharp P_{XY} = P_Y \otimes P_Z, \tilde{f} : \mathcal{X} \times \mathcal{Y} \ni (x, y) \mapsto (y, f(x, y))\}.$$

To see this, observe that if $f_{y\sharp} P_{X|y} = P_Z$, P_Y-a.e., then

$$P_Y \otimes P_Z(E \times F) = \int_E f_{y\sharp} P_{X|y}(F) dP_Y(y).$$

for any measurable rectangle $E \times F \subset \mathcal{Y} \times \mathcal{Z}$. On the other hand, if the above equality holds, then

$$\int_{\mathcal{Y}} P_Z(F) dP_Y(y) = P_Y \otimes P_Z(\mathcal{Y} \times F) = \int_{\mathcal{Y}} f_{y\sharp} P_{X|y}(F) dP_Y(y)$$

for any event $F \subset \mathcal{Z}$, yielding $f_{y\sharp} P_{X|y} = P_Z$, P_Y-a.e. Now observe that

$$\begin{aligned}
\int_E f_{y\sharp} P_{X|y}(F) dP_Y(y) &= \int_E P_{X|y}(f_y^{-1}(F)) dP_Y(y) \\
&= \int_E P_{X|y}(\{x \in \mathcal{X} \mid f(x, y) \in F\}) dP_Y(y) \\
&= P_{XY}(\{(x, y) \in \mathcal{X} \times \mathcal{Y} \mid y \in E, f(x, y) \in F\}) \\
&= P_{XY}(\{(x, y) \in \mathcal{X} \times \mathcal{Y} \mid \tilde{f}(x, y) \in E \times F\}) = \tilde{f}_\sharp P_{XY}(E \times F).
\end{aligned}$$

□

A.3. Proof of Corollary 3.2

Proof. Let

$$S = Y_{1:(n-1)} \in \mathcal{S} := \mathcal{Y}_{1:(n-1)},\ P_S = \bigotimes_{i=1}^{n-1} P_{Y_i},\ g(s, y_n, z_n) = g_1(y_1, \dots g_{n-1}(y_{n-1}, g_n(y_n, z_n)))$$

for $s = (y_1, \dots, y_{n-1})$. (7) reduces to

$$W_p^p(P_{XS}, P_{GS}) = \inf_{(h,\breve{h})\in\mathcal{H}} \mathbb{E}_{P_{XS}}\, d^p\big(X, g \circ (\mathcal{A}(h, \breve{h}))(X, S)\big) \quad \text{where}$$
$$\mathcal{H} = \{(h, \breve{h}) | h : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}_n, \breve{h} : \mathcal{X} \times \mathcal{S} \to \mathcal{Z}_n \text{ measurable}, (h, \breve{h})_\sharp P_{XS} = P_{Y_n} \otimes P_{Z_n}\},$$

and (8) also reduces to

$$W_p^p(P_{XS}, P_{GS}) = \inf_{(f,h)\in\mathcal{I}} \mathbb{E}_{P_{XS}}\, d^p\big(X, g \circ (\mathcal{A}_1 f) \circ (\mathcal{A}_0 h)(X, S)\big) \quad \text{where}$$
$$\mathcal{I} = \{(f, h) | f \colon \mathcal{X} \times \mathcal{S} \times \mathcal{Y}_n \to \mathcal{Z}_n, h \colon \mathcal{X} \times \mathcal{S} \to \mathcal{Y}_n \text{ measurable}, ((\mathcal{A}_1 f) \circ (\mathcal{A}_0 h))_\sharp P_{XS} = P_S \otimes P_{Y_n} \otimes P_{Z_n}\}.$$

Fix $(f, h) \in \mathcal{I}$. Define $\breve{h}(x, s) := f(x, s, h(x, s))$. Then $(\mathcal{A}_1 f) \circ (\mathcal{A}_0 h)(x, s) = (s, h(x, s), f(x, s, h(x, s))) = (s, h(x, s), \breve{h}(x, s))$. $((\mathcal{A}_1 f) \circ (\mathcal{A}_0 h))_\sharp P_{XS} = P_S \otimes P_{Y_n} \otimes P_{Z_n}$ implies $(h, \breve{h})_\sharp P_{XS} = P_{Y_n} \otimes P_{Z_n}$.

For $n = 1$, the proof is straightforward by omitting S. □

B. Stacked Generative Model with Correlated Response and Nuisance Variables

Let $S = Y_{1:(n-1)}$, the set of all nuisance variables, as in Appendix A.3. In this section, Y and S are allowed to be correlated. Figure 1d actually describes the *conditional* distribution $P_{X|S}$ of X given S.

In case the data are fully observed, the following result is obtained from Theorem 3.1 by considering $G = g_s(y, z_2)$ for $g_s = g_1(s, g_2(\cdot, \cdot))$ and $W_p^p(P_{XY|s}, P_{GY|s})$, $s \in \mathcal{S}$.

Corollary B.1. *If the conditional distributions $P_{XY|S}$ are non-atomic and g_1, g_2 are injective, then*

$$W_p^p(P_{XY|S}, P_{GY|S}) = \inf_{f\in\mathcal{F}} \mathbb{E}_{P_{XY|S}}\, d^p(X, g \circ (\mathcal{A}_1 f)(X, S, Y)),\ P_S\text{-a.e.} \quad \textit{where} \tag{18}$$
$$\mathcal{F} = \{f : \mathcal{X} \times \mathcal{S} \times \mathcal{Y} \to \mathcal{Z}_2 \textit{ measurable} \mid f_{ys} = f(\cdot, s, y), f_{ys\sharp} P_{X|ys} = P_{Z_2},\ P_{Y|s}\text{-a.e.}\}$$
$$= \{f : \mathcal{X} \times \mathcal{S} \times \mathcal{Y} \to \mathcal{Z}_2 \textit{ measurable} \mid \tilde{f}_{s\sharp} P_{XY|S} = P_{Y|S} \otimes P_{Z_2}, \tilde{f}_s : (x, y) \mapsto (y, f(x, s, y))\}.$$

If the response Y is missing, Theorem 2.1 applies directly if we replace Z with (Y, Z_2) and condition on S: $P_{G|s} = g_{s\sharp}(P_{Y|s} \otimes P_{Z_2})$ and hence

$$W_p^p(P_{X|S}, P_{G|S}) = \inf_{(h,\breve{h})\in\mathcal{H}} \mathbb{E}_{P_{X|S}}\, d^p(X, g \circ (\mathcal{A}_1(h, \breve{h}))(X, S)), P_S\text{-a.e.}, \quad \text{where}$$
$$\mathcal{H} = \big\{(h, \breve{h}) \mid h : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}, \breve{h} : \mathcal{X} \times \mathcal{S} \to \mathcal{Z}_2 \text{ measurable},$$
$$(h_s, \breve{h}_s)_\sharp P_{X|s} = P_{Y|s} \otimes P_{Z_2}, P_S\text{-a.e.}, h_s = h(\cdot, s), \breve{h}_s = \breve{h}(\cdot, s), s \in \mathcal{S}\big\}.$$

The following result connects this quantity with Corollary B.1.

Theorem B.2. *Under the conditions of Corollary B.1, the following holds for $g_1 : \mathcal{S} \times \mathcal{Z}_1 \to \mathcal{X}$ and $g_2 : \mathcal{Y} \times \mathcal{Z}_2 \to \mathcal{Z}_1$ that are both measurable and injective.*

$$W_p^p(P_{X|S}, P_{G|S}) = \inf_{(f,h)\in\mathcal{I}} \mathbb{E}_{P_{X|S}}\, d^p(X, g \circ (\mathcal{A}_1 f) \circ (\mathcal{A}_0 h)(X, S)), P_S\textit{-a.e.}, \quad \textit{where} \tag{19}$$
$$\mathcal{I} = \{(f, h) \mid f : \mathcal{X} \times \mathcal{Y} \times \mathcal{S} \to \mathcal{Z}_2, h : \mathcal{X} \times \mathcal{S} \to \mathcal{Y} \textit{ measurable}, (\tilde{f}_s \circ \tilde{h}_s)_\sharp P_{X|s} = P_{Y|s} \otimes P_{Z_2}, P_S\textit{-a.e.},$$
$$\tilde{f}_s : (x, y) \mapsto (y, f(x, s, y)), \tilde{h}_s : x \mapsto (x, h(x, s)),\ s \in \mathcal{S}\}.$$

Proof. Fix $(f,h) \in \mathcal{I}$. Define $\breve{h}_s(x) = f(x,s,h(x,s))$ and $\breve{h}(x,s) = \breve{h}_s(x)$. Then $\tilde{f}_s \circ \tilde{h}_s(x) = (h_s(x), \breve{h}_s(x))$ and hence $(h_s, \breve{h}_s)_\sharp P_{X|s} = P_{Y|s} \otimes P_{Z_2}$. This implies

$$\inf_{(h,\breve{h})\in\mathcal{H}} \mathbb{E}_{P_{X|S}}\, d^p(X, g_1(S, g_2(h(X,S), \breve{h}(X,S)))) \geq \inf_{(f,h)\in\mathcal{A}} \mathbb{E}_{P_{X|S}}\, d^p(X, g_1(S, g_2(h(X,S), f(X,S,h(X,S)))))).$$

For the opposite direction, fix $(h,\breve{h}) \in \mathcal{H}$. Define $f_s : (x,y) \mapsto \breve{h}_s(x)$ and $f : (x,s,y) \mapsto f_s(x,y)$. Also define $\tilde{f}_s(x,y) = (y, f(x,s,y))$ and $\tilde{h}_s(x) = (x, h(x,s))$. Then $(h_s(x), \breve{h}_s(x)) = \tilde{f}_s \circ \tilde{h}_s(x)$ and thus $(\tilde{f}_s \circ \tilde{h}_s)_\sharp P_{X|s} = P_{Y|s} \otimes P_{Z_2}$ and $(f,h) \in \mathcal{I}$. □

Building StrWAEs. The constraints defining the encoder set $\mathcal{F}$ in Corollary B.1 are equivalent to $f(X,Y,S) \overset{d}{=} Z_2$ and $Y \perp\!\!\!\perp f(X,Y,S)$ given S. Proceeding as Section 3.2.1 while considering the additional independence constraint $Z_1 \perp\!\!\!\perp S$, a penalized version of (18) is

$$\begin{aligned}\inf_f \big\{ &\mathbb{E}_{P_S}[\mathbb{E}_{P_{XY|S}}\, d^p(X, g \circ (\mathcal{A}_1 f)(X,S,Y)) + \lambda \mathcal{D}(f_{S\sharp} P_{XY|S} \| P_{Z_2}) \\ &+ \mu_1 \mathcal{H}(\{Y|S\}, \{f(X,Y,S)|S\})] + \mu_2 \mathcal{H}(S, g_2(Y, f(X,Y,S)))\big\}\end{aligned}$$

for appropriate $\lambda, \mu_1, \mu_2 > 0$ where $f_s(x,y) := f(x,s,y)$. If Y is only partially observed, Theorem B.2 entails a functional

$$\begin{aligned}\delta(f,h,g_1,g_2) = &\mathbb{E}_{P_S}[\mathbb{E}_{P_{X\hat{Y}_h|S}}\, d^p(X, g \circ (\mathcal{A}_1 f)(X,S,\hat{Y}_h)) + \lambda_1 \mathcal{D}(f_{S\sharp} P_{XY|S} \| P_{Z_2}) \\ &+ \lambda_2 \mathcal{D}(P_{\hat{Y}_h|S} \| P_{Y|S}) + \mu_1 \mathcal{H}(\{\hat{Y}_h|S\}, \{f(X,S,\hat{Y}_h)|S\})] + \mu_2 \mathcal{H}(S, g_2(\hat{Y}_h, f(X,S,\hat{Y}_h))\end{aligned}$$

to minimize, where $\hat{Y}_h = Y$ if Y is observed and $\hat{Y}_h = h(X,S)$ otherwise.

C. Ablation Study

We conducted an ablation study on the semi-supervised classification task to explore the sensitivity of hyperparameters. Recall that we used the sum of cross entropy (if Y is observed, 0 otherwise) and MMD for $\mathcal{D}(P_{\hat{Y}_h}\|P_Y)$ in (12). We summarised our analysis of the sensitivity of penalty coefficients as follows.

1. λ_1 (coefficient of the GAN penalty): the performance was good when the coefficient of the GAN penalty λ_1 was balanced to the scale of the GAN penalty with the reconstruction term. Moreover, for the other parameters, the model was well trained when we scaled the other terms such as HSIC and cross entropy smaller than the previous two terms.

2. λ_2 (coefficient of cross entropy and MMD): When the classification accuracy is based, conditional generation will be good, so the cross entropy term should be prioritized during the training. In Table 4, there is significant performance degradation when either the cross entropy or MMD term are missing. Our choice of λ_2 was 500 or 1000 to match the scale of the cross entropy to the reconstruction term after 1 epoch.

3. μ_1 (coefficient of HSIC): Table 4 shows that the conditional generation performance becomes worse if the HSIC term is missing. Note that HSIC is needed to ensure the independence within the encoder constraints, $(\hat{Y}_h, f(X, \hat{Y}_h)) \overset{d}{=} (Y, Z)$. A good rule of thumb for this is to first fit the marginal distributions and then ensure independence. Therefore, we set μ_1 to make the HSIC term smaller in scale.

Table 4: Ablation studies over penalty terms on MNIST and SVHN datasets. The bolded row refers to the penalty coefficients reported in Table 1. MMD: maximum mean discrepancy. CE: cross entropy. A/B: A for the MNIST dataset and B for the SVHN dataset.

$\mathcal{D}(P_{\hat{Y}_h}\|P_Y)$	λ_2	μ_1	MNIST(100)			SVHN(1000)		
			Classification	Conditional Generation		Classification	Conditional Generation	
				Y	$\hat{Y}$		Y	$\hat{Y}$
CE + MMD	**500/1000**	**100/10**	**98.69**	**96.06**	**94.74**	**85.59**	**98.80**	**82.69**
CE	500/1000	100/10	88.00	25.59	22.98	83.74	96.67	79.48
MMD	500/1000	100/10	12.85	51.13	34.12	14.16	21.49	11.39
CE + MMD	500/1000	0/0	98.48	92.89	91.42	76.35	95.39	73.29
CE	500/1000	0/0	87.70	32.78	32.74	83.00	94.80	77.46
MMD	100/1000	0/0	22.44	24.03	22.46	13.17	25.11	11.37
-	0/0	500/10	65.48	99.48	43.99	15.43	64.13	12.30
-	0/0	0/0	62.91	99.44	47.55	20.21	73.32	12.60

D. Additional Experiments on the Choice of Divergence

We experimented with the Kullback-Leibler (KL) and Sinkhorn divergences in addition to the GAN penalty in place of $\mathcal{D}(f_\sharp P_{X\hat{Y}_h}\|P_Z)$ in (12). Classification accuracy similar to Table 1 is presented in Table 5. Using Sinkhorn divergence achieved a similar classification performance, but the conditional generation performance dropped significantly. We used the normal distribution as the prior, but Patrini et al. (2020) reports that the Sinkhorn divergence for the normal distribution did not perform well on image data. We found that penalties that require adversarial training performed better empirically. To compute the KL divergence between the aggregate posterior and the prior distributions, we utilized a variational representation of KL divergence, named Donsker-Varadhan formula: $D_{KL}(P, Q) = \sup_{d:\mathcal{Z}\to\mathbb{R}} \mathbb{E}_P[d(X)] - \mathbb{E}_Q[e^{d(X)}]$.

Table 5: Classification accuracies of different divergences for the prior matching term on MNIST and SVHN datasets.

$\mathcal{D}(f_\sharp P_{X\hat{Y}_h}\|P_Z)$	MNIST(100)			SVHN(1000)		
	Classification	Conditional Generation		Classification	Conditional Generation	
		Y	$\hat{Y}$		Y	$\hat{Y}$
Jensen-Shannon (=GAN penalty)	98.69	96.06	94.74	85.59	98.80	82.69
KL	98.65	97.23	95.62	81.50	97.94	78.30
Sinkhorn	98.38	52.65	51.59	78.57	77.65	56.79

E. Further implementation details

The prior P_Z was set to be a normal distribution $\mathcal{N}(0, I_k)$, where k is the dimension of the latent space $\mathcal{Z}$. For the penalty divergences $\mathcal{D}$ between encoded Z and samples from P_Z, we used the GAN loss (Goodfellow et al., 2014), which requires an additional discriminator. Our models were trained using RAdam (Liu et al., 2020) and Adam (Kingma & Ba, 2014) optimizers.

E.1. Semi-supervised learning

The autoencoder network of StrWAE for the MNIST (resp. SVHN) dataset had 0.9M (resp. 6.8M) parameters, and the discriminator had 0.018M (resp. 0.035M) parameters; see Table 6 (resp. Table 7). We trained the model end-to-end with 500 and 200 epochs, respectively. For the dimension of Z, we used $k = 10$ for the MNIST dataset and $k = 20$ for the SVHN dataset. We set the hyper-parameters as follows: for the MNIST, $\lambda_1 = 100$, $\lambda_2 = 100$, $\mu_1 = 500$, and $\mu_2 = 0$ for the SVHN, $\lambda_1 = 10$, $\lambda_2 = 10$, $\mu_1 = 1000$, and $\mu_2 = 0$.

The similar architecture is used for the adversarial autoencoder (AAE, Makhzani et al., 2016) and the characteristic capturing VAE (CCVAE, Joy et al., 2021). For the AAE, the prior P_Z was set to $\mathcal{N}(0, I_k)$ and SGD with momentum is used for optimization. For the Smooth-ELBO (SHOT-VAE, Feng et al., 2021) and Gaussian mixture VAE (GMVAE, Dilokthanakul et al., 2016), the networks architecture proposed by the authors achieves better performance. To generate images conditional on specific digits, it is necessary to assign clusters to corresponding digits. For this purpose, we employed the labeled data used for semi-supervised learning, despite it being unlabeled during GMVAE training. The encoder deduced the cluster to which a digit image belongs, and the cluster that was most frequently inferred for a particular digit image was designated as the representative cluster for that digit.

Note that we utilized an extra set consisting of approximately 530K images from the SVHN dataset for training.

E.2. Identity-preserving conditional generation

The face region of images from the VGGFace2 dataset (Cao et al., 2018) were cropped and resized into a size of 128×128. The autoencoder architecture of the StrWAE had 88.4M parameters, and the discriminator architecture had 206k parameters (Table 8). We pre-trained the (Y, S)-encoder with 3K iterations, then optimized the rest of the network for 30K iterations. The results were compared with Fader Network (Lample et al., 2018) having an encoder-decoder architecture with 70.2M parameters and a discriminator architecture with 0.48M parameters (Table 9) trained for 20K iterations. In our experiment, we set $\lambda_1 = 100, \mu_1 = 10000$, and $\lambda_2 = \mu_2 = 0$.

E.3. Invariant representations

The cropped version of the Extended Yale Face Database B dataset (Georghiades et al., 2001; Lee et al., 2005) were resized into a size of 128×128. The autoencoder architecture of the StrWAE had total of 17.4M parameters, and the discriminator architecture had 881 parameters (Table 10). After pre-training the $h(X)$ in (14) with 2K iterations, we optimized the network for 5K iterations. The results were compared with the VAE and HSIC-constrained VAE (HCV, Lopez et al., 2018). Here, VAE denotes HCV without HSIC penalty term. To compare with our model, HCV was trained to minimize a semi-supervised objective. For the Extended Yale Face Database B dataset, we set $\lambda_1 = 100, \mu_1 = 10, \mu_2 = 50000000$, and $\lambda_2 = 0$.

Computing infrastructure. We trained the networks with Intel® Xeon® CPU Silver 4114 @ 2.20GHz processors and Nvidia Titan V GPUs with 12GB memory. For the VGGFace2 experiments, we trained the network using four GPUs; those for the other experiments were all trained using a single GPU. All the implementations were based on Python 3.11, PyTorch 2.1.1, and CUDA 12.1.

F. Additional Figures

MNIST and SVHN. Figure 6 depicts the class-preserving samples trained with MNIST and SVHN dataset. Note that all test data were not used in training, and we did not provide information about actual digit class Y of the test data to generate images. Figure 7 presents the full result of the style transfer task shown in Figure 3.

Table 6: Network architecture for the MNIST dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
f	1	Convolution	32	4x4	2x2	Yes	LeakyReLU	X
	2	Convolution	64	4x4	2x2	Yes	LeakyReLU	1
	3	Convolution	128	4x4	2x2	Yes	LeakyReLU	2
	4	Linear	4x4x128	-	-	No	LeakyReLU	3
	5	Linear	128	-	-	No	LeakyReLU	4
	6	Linear	20	-	-	No	LeakyReLU	5
	h	Softmax	10	-	-	No	-	6
	7	Linear	10	-	-	No	-	(6,Y) or (6,h)
g	1	Linear	128	-	-	No	LeakyReLU	(Y,f) or (h,f)
	2	Linear	128	-	-	No	LeakyReLU	1
	3	Linear	4x4x128	-	-	No	LeakyReLU	2
	4	Transposed Convolution	64	4x4	2x2	Yes	LeakyReLU	3
	5	Transposed Convolution	32	4x4	2x2	Yes	LeakyReLU	4
	6	Transposed Convolution	1	4x4	2x2	No	Sigmoid	5
Discriminator	1	Linear	128	-	-	No	LeakyReLU	f
	2	Linear	128	-	-	No	LeakyReLU	1
	3	Linear	1	-	-	No	-	2

Table 7: Network architecture for the SVHN dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
f	1	Convolution	64	4x4	2x2	Yes	LeakyReLU	X
	2	Convolution	128	4x4	2x2	Yes	LeakyReLU	1
	3	Convolution	256	4x4	2x2	Yes	LeakyReLU	2
	4	Convolution	512	4x4	2x2	Yes	LeakyReLU	3
	5	Linear	256	-	-	Yes	LeakyReLU	4
	6	Linear	256	-	-	Yes	LeakyReLU	5
	7	Linear	256	-	-	Yes	LeakyReLU	6
	7	Linear	256	-	-	Yes	LeakyReLU	7
	h	Softmax	10	-	-	No	-	8
	9	Linear	20	-	-	No	-	9
g	1	Linear	256	-	-	Yes	LeakyReLU	(Y,f) or (h,f)
	2	Linear	256	-	-	Yes	LeakyReLU	1
	3	Linear	256	-	-	Yes	LeakyReLU	2
	4	Linear	2x2x512	-	-	Yes	LeakyReLU	3
	5	Transposed Convolution	256	4x4	2x2	Yes	LeakyReLU	4
	6	Transposed Convolution	128	4x4	2x2	Yes	LeakyReLU	5
	7	Transposed Convolution	64	4x4	2x2	Yes	LeakyReLU	6
	8	Transposed Convolution	1	4x4	2x2	No	Sigmoid	7
Discriminator	1	Linear	128	-	-	No	LeakyReLU	f
	2	Linear	128	-	-	No	LeakyReLU	1
	3	Linear	128	-	-	No	LeakyReLU	2
	5	Linear	1	-	-	No	-	3

Table 8: StrWAE architecture for the VGGFace2 dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
(h_1, h_2)	1	Convolution	128	5x5	2x2	Yes	ReLU	X
	2	Convolution	256	5x5	2x2	Yes	ReLU	1
	3	Convolution	512	5x5	2x2	Yes	ReLU	2
	4	Convolution	1024	5x5	2x2	Yes	ReLU	3
	5	Linear	71	-	-	-	-	4
f	1	Convolution	128	5x5	2x2	Yes	ReLU	X
	2	Convolution	128	5x5	1x1	Yes	ReLU	1
	3	Convolution	256	5x5	2x2	Yes	ReLU	2
	4	Convolution	256	5x5	1x1	Yes	ReLU	3
	5	Convolution	512	5x5	2x2	Yes	ReLU	4
	6	Convolution	512	3x3	1x1	Yes	ReLU	5
	7	Convolution	1024	3x3	2x2	Yes	ReLU	6
	8	Convolution	1024	3x3	1x1	Yes	ReLU	7
	9	Linear	32	-	-	-	-	8
g	1	Linear	8x8x1024	-	-	No	-	h_1, h_2, f
	2	Transposed Convolution	512	5x5	2x2	Yes	LeakyReLU	1
	3	Residual Block	512	5x5, 1x1	1x1	Yes	LeakyReLU	2
	4	Transposed Convolution	256	5x5	2x2	Yes	LeakyReLU	3
	5	Residual Block	256	5x5, 1x1	1x1	Yes	LeakyReLU	4
	6	Transposed Convolution	128	5x5	2x2	Yes	LeakyReLU	5
	7	Residual Block	128	3x3, 1x1	1x1	Yes	LeakyReLU	6
	8	Transposed Convolution	64	5x5	2x2	Yes	LeakyReLU	7
	9	Residual Block	64	3x3, 1x1	1x1	Yes	LeakyReLU	8
	10	Convolution	3	3x3	1x1	No	Sigmoid	9
Discriminator	1	Linear	256	-	-	No	ReLU	f
	2	Linear	256	-	-	No	ReLU	1
	3	Linear	256	-	-	No	ReLU	2
	4	Linear	256	-	-	No	ReLU	3
	5	Linear	1	-	-	No	-	4

Table 9: Fader Network architecture for the VGGFace2 dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
Encoder f	1	Convolution	128	5x5	2x2	Yes	ReLU	X
	2	Convolution	128	5x5	1x1	Yes	ReLU	1
	3	Convolution	256	5x5	2x2	Yes	ReLU	2
	4	Convolution	256	5x5	1x1	Yes	ReLU	3
	5	Convolution	512	5x5	2x2	Yes	ReLU	4
	6	Convolution	512	3x3	1x1	Yes	ReLU	5
	7	Convolution	1024	3x3	2x2	Yes	ReLU	6
	8	Convolution	1024	3x3	1x1	Yes	ReLU	7
	9	Linear	96	-	-	-	-	8
Decoder g	1	Linear	8x8x1024	-	-	No	-	S, Z
	2	Transposed Convolution	512	5x5	2x2	Yes	LeakyReLU	1
	3	Residual Block	512	5x5, 1x1	1x1	Yes	LeakyReLU	2
	4	Transposed Convolution	256	5x5	2x2	Yes	LeakyReLU	3
	5	Residual Block	256	5x5, 1x1	1x1	Yes	LeakyReLU	4
	6	Transposed Convolution	128	5x5	2x2	Yes	LeakyReLU	5
	7	Residual Block	128	3x3, 1x1	1x1	Yes	LeakyReLU	6
	8	Transposed Convolution	64	5x5	2x2	Yes	LeakyReLU	7
	9	Residual Block	64	3x3, 1x1	1x1	Yes	LeakyReLU	8
	10	Convolution	3	3x3	1x1	No	Sigmoid	9
Discriminator	1	Linear	384	-	-	No	ReLU	Z
	2	Linear	384	-	-	No	ReLU	1
	3	Linear	384	-	-	No	ReLU	2
	4	Linear	384	-	-	No	ReLU	3
	5	Linear	7	-	-	No	-	4

Table 10: StrWAE architecture for the Extended Yale B dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
h	1	Convolution	64	5x5	2x2	Yes	ReLU	X
	2	Convolution	128	5x5	2x2	Yes	ReLU	1
	3	Convolution	256	5x5	2x2	Yes	ReLU	2
	4	Convolution	512	3x3	2x2	Yes	ReLU	3
	5	Convolution	1024	3x3	2x2	Yes	ReLU	4
	6	Linear	8	-	-	-	-	5
f	1	Convolution	32	5x5	2x2	Yes	ReLU	X
	2	Convolution	64	5x5	2x2	Yes	ReLU	1
	3	Convolution	128	5x5	2x2	Yes	ReLU	2
	4	Convolution	256	3x3	2x2	Yes	ReLU	3
	5	Convolution	512	3x3	2x2	Yes	ReLU	4
	6	Linear	2	-	-	-	-	5
g_1	1	Linear	8x8x1024	-	-	No	-	g_2
	2	Transposed Convolution	512	3x3	2x2	Yes	LeakyReLU	1
	3	Transposed Convolution	256	3x3	2x2	Yes	LeakyReLU	2
	4	Convolution	256	3x3	1x1	Yes	LeakyReLU	3
	5	Transposed Convolution	128	5x5	2x2	Yes	LeakyReLU	4
	6	Transposed Convolution	64	5x5	2x2	Yes	LeakyReLU	5
	7	Convolution	64	5x5	1x1	Yes	LeakyReLU	6
	8	Convolution	1	5x5	1x1	No	Sigmoid	7
g_2	1	Linear	50	-	-	Yes	LeakyReLU	h, f
Discriminator	1	Linear	16	-	-	No	ReLU	f
	2	Linear	16	-	-	No	ReLU	1
	3	Linear	16	-	-	No	ReLU	2
	4	Linear	16	-	-	No	ReLU	3
	5	Linear	1	-	-	No	-	4

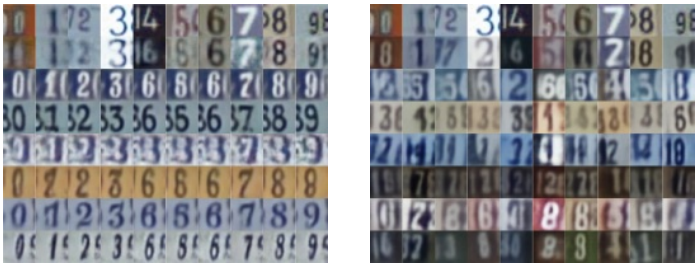
Table 11: HCV architecture for the Extended Yale B dataset.

Map	Layer	Operation	Filters	Kernel	Strides	Batch norm	Activation	Linked layer
$Q_{Z_1\|X,S}$	1	Convolution	64	5x5	2x2	Yes	ReLU	X
	2	Convolution	128	5x5	2x2	Yes	ReLU	1
	3	Convolution	256	5x5	2x2	Yes	ReLU	2
	4	Convolution	512	3x3	2x2	Yes	ReLU	3
	5	Convolution	1024	3x3	2x2	Yes	ReLU	4
	μ	Linear	10	-	-	No	-	5, S
	σ^2	Linear	10	-	-	No	-	5, S
	Output $(Z_1\|X,S)$	Sample $Z_1\|X,S$	-	-	-	-	-	μ, σ^2
$Q_{Z_2\|Z_1,Y}$	1	Linear	20	-	-	Yes	ReLU	Z_1, Y
	μ	Linear	10	-	-	No	-	1
	σ^2	Linear	10	-	-	No	-	1
	Output $(Z_2\|Z_1,Y)$	Sample $Z_2\|Z_1,Y$	-	-	-	-	-	μ, σ^2
$Q_{Y\|Z_1}$	1	Linear	20	-	-	Yes	ReLU	Z_1
	2	Linear	38	-	-	No	-	1
$P_{Z_1\|Z_2,Y}$	1	Linear	20	-	-	Yes	ReLU	Z_2, Y
	μ	Linear	10	-	-	No	-	1
	σ^2	Linear	10	-	-	No	-	1
	Output $(Z_1\|Z_2,Y)$	Sample $Z_1\|Z_2,Y$	-	-	-	-	-	μ, σ^2
$P_{X\|Z_1,S}$	1	Linear	8x8x1024	-	-	No	-	Z_1, S
	2	Transposed Convolution	512	3x3	2x2	Yes	ReLU	1
	3	Transposed Convolution	256	3x3	2x2	Yes	ReLU	2
	4	Convolution	256	3x3	1x1	Yes	ReLU	3
	5	Transposed Convolution	128	5x5	2x2	Yes	ReLU	4
	6	Transposed Convolution	64	5x5	2x2	Yes	ReLU	5
	7	Convolution	64	5x5	1x1	Yes	ReLU	6
	8	Convolution	1	5x5	1x1	No	Sigmoid	7

(a) StrWAE (b) AAE

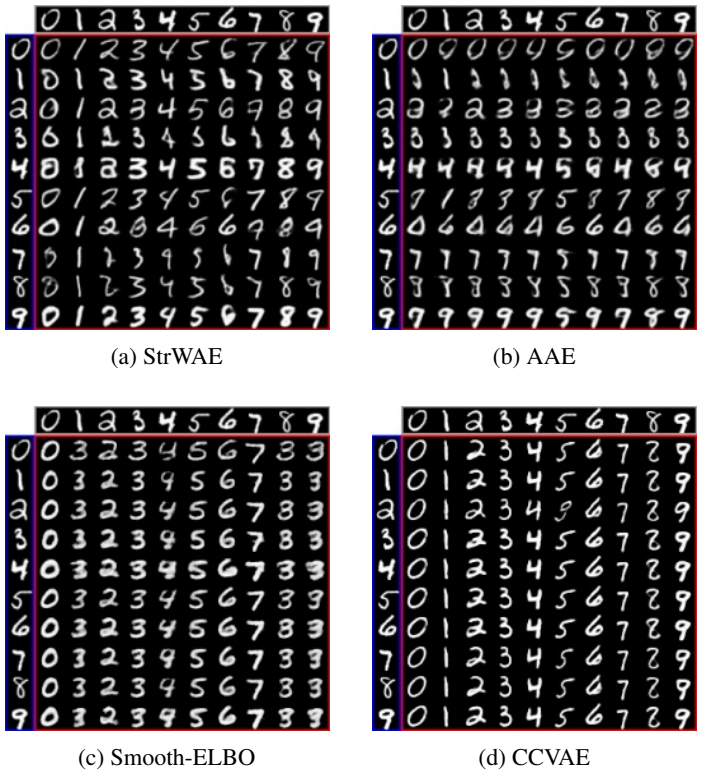
(c) Smooth-ELBO (d) CCVAE

(e) StrWAE (f) AAE



(g) Smooth-ELBO (h) CCVAE

Figure 6: Class-preserving generation from the MNIST and SVHN dataset. Green box and red box denote reconstruction images and conditionally generated images respectively.



(a) StrWAE (b) AAE

(c) Smooth-ELBO (d) CCVAE

Figure 7: Style transfer in the MNIST dataset.

VGGFace2. Figure 8 shows the class-preserving generation results for the VGGFace2 dataset, extending the corresponding results in Figure 2. Figure 9 shows the extended style transfer results corresponding to Figure 3. Figures 10 and 11 extend the attribute manipulation results (Figure 4), including gender, sunglasses and mouth open state. Manipulating the gender attribute changed eyes and lips of the test data so that the images resemble stereotypical male (or female) pictures. Letting the sunglasses attribute positive produced decoded images having darkened eye area that resembles sunglasses; making it negative on an image with sunglasses produced one without them. Manipulating mouth open could make the manipulated images with either mouth wide open or closed.

Extended Yale B. Figure 12 depicts the class-preserving samples generated from StrWAE and HCV, trained with the Extended Yale B dataset, which enlarges the corresponding results presented in Figure 2. Note that StrWAE has better generation quality and clearer effects of manipulating the lighting conditions than HCV, while keeping the identity of the test data.

G. Experiments on a Tabular Dataset

We reproduced the experiment on fair representations in Louizos et al. (2016) using the Adult Income dataset, where the response variable Y is an indicator of high income and the nuisance variable S is gender. The size of train data and test data are 30,162 and 15,060, respectively, and all variables are fully-observed. Continuous variables were categorized into 5 categories, and one-hot encoding was used for variables with multiple category. The goal here is that the fair representation $Z_1 = g_2(Y, f(X, Y, S))$ encompasses information of Y while excluding information related to S. Consequently, we expect this representation to yield high classification accuracy for Y and low classification accuracy for S. Furthermore, the probability of predicting $Y = 1$ should be similar regardless of S. To formulate our framework in this setting, modeling of the conditional distribution of the data on S is required since Y and S are correlated, in which case the formulation in Appendix B should be applied. Fairness is measured by the demographic parity, $\Delta_{DP} := |P(\hat{Y} = 1|S = 1) - P(\hat{Y} = 1|S = 0)|$. We used the logistic regression to predict Y and S from Z_1. StrWAE showed higher accuracy on Y and lower Δ_{DP} compared to HCV. VAE achieved the highest accuracy on Y and lowest accuracy on S, but with large statistical parity.

Figure 8: Class-preserving generation from the VGGFace2 dataset.

Table 12: Fair representation results. Accuracy is measured by logistic regression. VAE denotes HCV without HSIC penalty term.

Model	Y-accuracy	S-accuracy	Δ_{DP}
VAE	**0.8249 (±7.818e-4)**	**0.6638 (±5.458e-4)**	0.1586 (±3.311e-3)
HCV	0.8090 (±1.972e-3)	0.6738 (±0.0)	0.0718 (±4.748e-3)
StrWAE	0.8193 (±6.567e-3)	0.6735 (±4.384e-4)	**0.05583 (± 0.01200)**

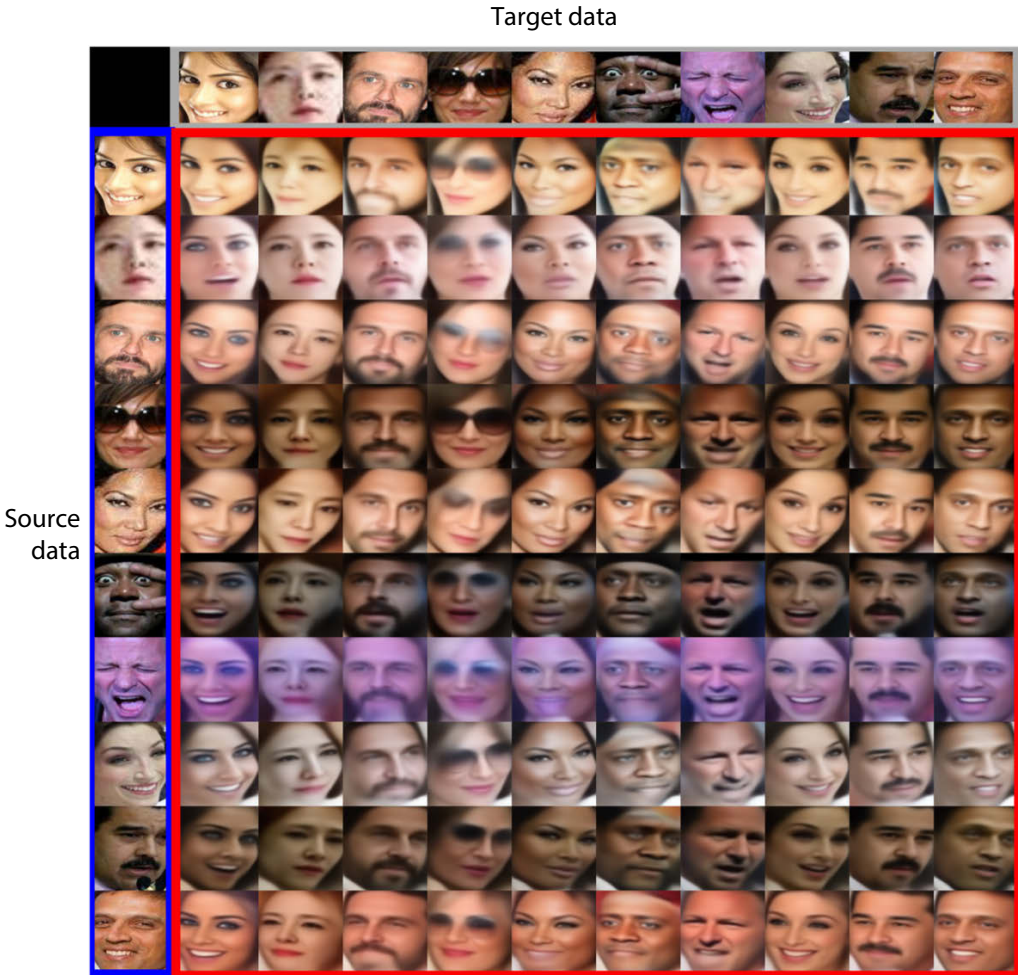


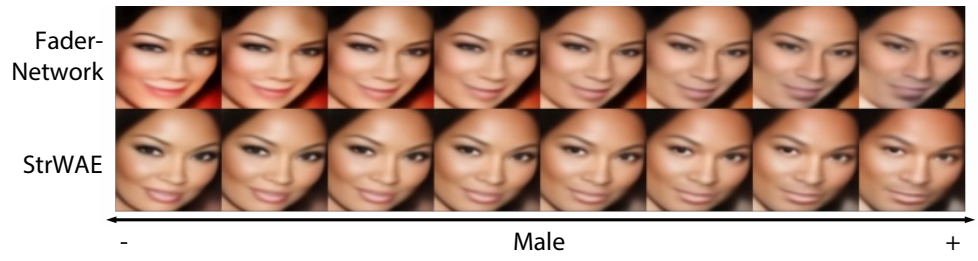
Figure 9: Style transfer in the VGGFace2 dataset.

(a) Attribute Manipulation - Male

(b) Attribute Manipulation - Sunglasses

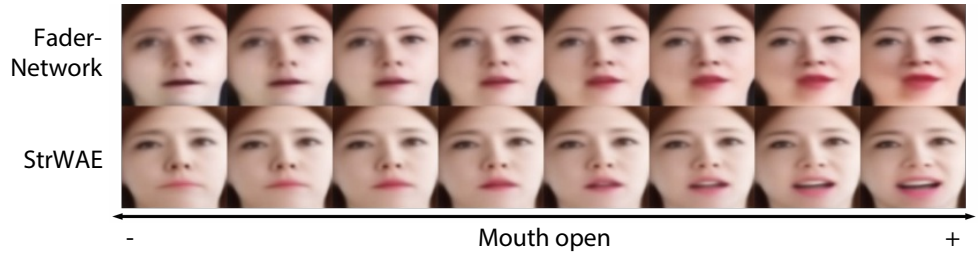
(c) Attribute Manipulation - Mouth open

Figure 10: Decoded images with attribute score manipulated to either 4.0 (red box, first row) or -3.0 (blue box, third row).



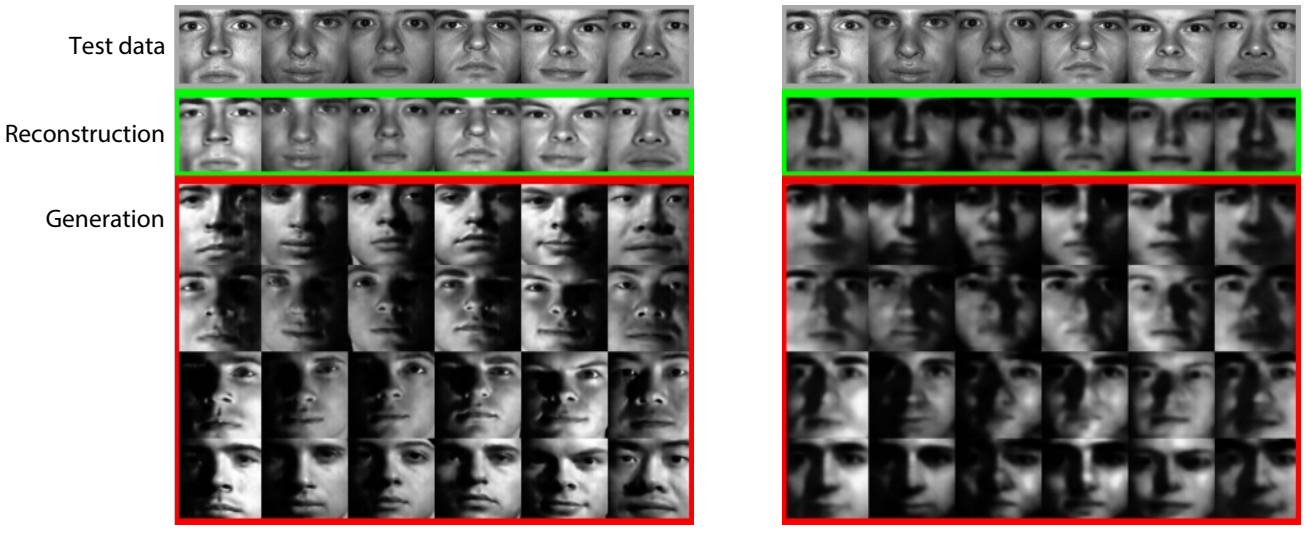
(a) Attribute Interpolation - Male

(b) Attribute Interpolation - Sunglasses



(c) Attribute Interpolation - Mouth open

Figure 11: Decoded images with attribute score interpolated in trained Fader Network and StrWAE



(a) StrWAE (b) HCV

Figure 12: Class-preserving generation from the Extended Yale B dataset.