Learning Autoencoders with Relational Regularization

Hongteng Xu [* 1 2] **Dixin Luo** [* 2] **Ricardo Henao** [2] **Svati Shah** [2] **Lawrence Carin** [2]

Abstract

A new algorithmic framework is proposed for learning autoencoders of data distributions. We minimize the discrepancy between the model and target distributions, with a *relational regularization* on the learnable latent prior. This regularization penalizes the fused Gromov-Wasserstein (FGW) distance between the latent prior and its corresponding posterior, allowing one to flexibly learn a structured prior distribution associated with the generative model. Moreover, it helps co-training of multiple autoencoders even if they have heterogeneous architectures and incomparable latent spaces. We implement the framework with two scalable algorithms, making it applicable for both probabilistic and deterministic autoencoders. Our relational regularized autoencoder (RAE) outperforms existing methods, *e.g.*, the variational autoencoder, Wasserstein autoencoder, and their variants, on generating images. Additionally, our relational co-training strategy for autoencoders achieves encouraging results in both synthesis and real-world multi-view learning tasks. The code is at https://github.com/HongtengXu/Relational-AutoEncoders.

1. Introduction

Autoencoders have been used widely in many challenging machine learning tasks for generative modeling, *e.g.*, image (Kingma & Welling, 2013; Tolstikhin et al., 2018) and sentence (Bowman et al., 2016; Wang et al., 2019) generation. Typically, an autoencoder assumes that the data in the sample space $\mathcal{X}$ may be mapped to a low-dimensional manifold, which can be represented in a latent space $\mathcal{Z}$. The autoencoder fits the unknown data distribution p_x via

---

[*]Equal contribution [1]Infinia ML Inc., Durham, NC, USA [2]Duke University, Durham, NC, USA. Correspondence to: Hongteng Xu <hongtengxu313@gmail.com>, Dixin Luo <dixin.luo@duke.edu>.

a latent-variable model denoted p_G, specified by a prior distribution p_z on latent code $z \in \mathcal{Z}$ and a generative model $G : \mathcal{Z} \mapsto \mathcal{X}$ mapping the latent code to the data $x \in \mathcal{X}$. Learning seeks to minimize the discrepancy between p_x and p_G. According to the choice of the discrepancy, we can derive different autoencoders. For example, the variational autoencoder (Kingma & Welling, 2013) applies the KL-divergence as the discrepancy and learns a probabilistic autoencoder via maximizing the evidence lower bound (ELBO). The Wasserstein autoencoder (WAE) (Tolstikhin et al., 2018) minimizes a relaxed form of the Wasserstein distance between p_x and p_G, and learns a deterministic autoencoder. In general, the objective function approximating the discrepancy consists of a reconstruction loss of observed data and a regularizer penalizing the difference between the prior distribution p_z and the posterior derived by encoded data, *i.e.*, $q_{z|x}$. Although existing autoencoders have achieved success in many generative tasks, they often suffer from the following two problems.

Regularizer misspecification Typical autoencoders, like the VAE and WAE, fix the p_z as a normal distribution, which often leads to the problem of over-regularization. Moreover, applying such an unstructured prior increases the difficulties in conditional generation tasks. To avoid oversimplified priors, the Gaussian mixture VAE (GMVAE) (Dilokthanakul et al., 2016) and the VAE with VampPrior (Tomczak & Welling, 2018) characterize their priors as learnable mixture models. However, without side information (Wang et al., 2019), jointly learning the autoencoder and the prior suffers from a high risk of under-regularization, which is sensitive to the setting of hyperparameters (*e.g.*, the number of mixture components and the initialization of the prior).

Co-training of heterogeneous autoencoders Solving a single task often relies on the data in different domains (*i.e.*, multi-view data). For example, predicting the mortality of a patient may require both her clinical record and genetic information. In such a situation, we may need to learn multiple autoencoders to extract latent variables as features from different views. Traditional multi-view learning strategies either assume that the co-trained autoencoders share the same latent distributions (Wang et al., 2015; Ye et al., 2016), or assume that there exists an explicit transform between different latent spaces (Wang et al., 2016). These assumptions are questionable in practice, as the correspond-

ing autoencoders can have heterogeneous architectures and incomparable latent spaces. How to co-train such heterogeneous autoencoders is still an open problem.

To overcome the aforementioned problems, we propose a new **R**elational regularized **A**uto**E**ncoder (RAE). As illustrated in Figure 1(a), we formulate the prior p_z as a Gaussian mixture model. Differing from existing methods, however, we leverage the Gromov-Wasserstein (GW) distance (Mémoli, 2011) to regularize the structural difference between the prior and the posterior in a relational manner, *i.e.*, comparing the distance between samples from the prior with samples from the posterior, and restricting their difference. Considering this relational regularizer allows us to implement the discrepancy between p_z and $q_{z|x}$ as the fused Gromov-Wasserstein (FGW) distance (Vayer et al., 2018a). Besides imposing structural constraints on the prior distribution within a single autoencoder, for multiple autoencoders with different latent spaces (*e.g.*, the 2D and 3D latent spaces shown in Figure 1(b)) we can train them jointly by applying the relational regularizer to their posterior distributions.

The proposed relational regularizer is applicable for both probabilistic and deterministic autoencoders, corresponding to approximating the FGW distance as hierarchical FGW and sliced FGW, respectively. We demonstrate the rationality of these two approximations and analyze their computational complexity. Experimental results show that i) learning RAEs helps achieve structured prior distributions and also suppresses the under-regularization problem, outperforming related approaches in image-generation tasks; and ii) the proposed relational co-training strategy is beneficial for learning heterogeneous autoencoders, which has potential for multi-view learning tasks.

2. Relational Regularized Autoencoders

2.1. Learning mixture models as structured prior

Following prior work with autoencoders (Tolstikhin et al., 2018; Kolouri et al., 2018), we fit the model distribution p_G by minimizing its Wasserstein distance to the data distribution p_x, *i.e.*, $\min D_w(p_x, p_G)$. According to Theorem 1 in (Tolstikhin et al., 2018), we can relax the Wasserstein distance and formulate the learning problem as follows:

$$\min_{G,Q} \underbrace{\mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;Q}}[d(x, G(z))]}_{\text{reconstruction loss}} + \gamma \underbrace{D(\mathbb{E}_{p_x}[q_{z|x;Q}], p_z)}_{\text{distance(posterior, prior)}}, \quad (1)$$

where $G : \mathcal{Z} \mapsto \mathcal{X}$ is the target generative model (decoder); $q_{z|x;Q}$ is the posterior of z given x, parameterized by an encoder $Q : \mathcal{X} \mapsto \mathcal{Z}$; d represents the distance between samples; and D is an arbitrary discrepancy between distributions. Accordingly, $q_{z;Q} = \mathbb{E}_{p_x}[q_{z|x;Q}]$ is the marginal distribution derived from the posterior. Parameter γ achieves

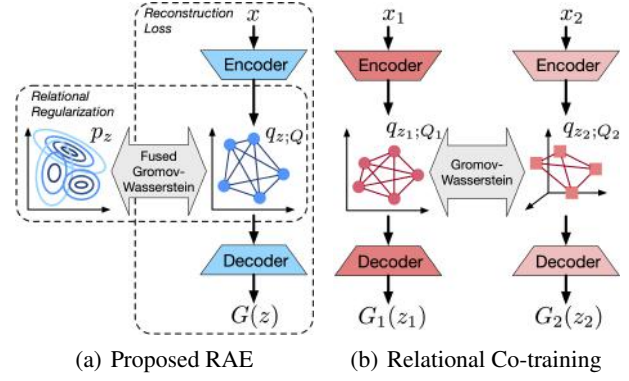(a) Proposed RAE   (b) Relational Co-training

Figure 1. (a) Learning a single autoencoder with relational regularization. (b) Relational co-training of the autoencoders with incomparable latent spaces.

a trade-off between reconstruction loss and the regularizer.

Instead of fixing p_z as a normal distribution, we seek to learn a structured prior associated with the autoencoder:

$$\min_{G,Q,p_z\in\mathcal{P}} \mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;Q}}[d(x, G(z))] + \gamma D(q_{z;Q}, p_z). \quad (2)$$

where $\mathcal{P}$ is the set of valid prior distributions, which is often assumed as a set of (Gaussian) mixture models (Dilokthanakul et al., 2016; Tomczak & Welling, 2018). Learning the structured prior allows one to explore the clustering structure of the data and achieve conditional generation (*i.e.*, sampling latent variables from a single component of the prior and generating samples accordingly).

2.2. Relational regularization via Gromov-Wasserstein

Jointly learning the prior and the autoencoder may lead to under-regularization in the training phase – it is easy to fit p_z to $q_{z;Q}$ without harm to the reconstruction loss. Solving this problem requires introduction of structural constraints when comparing these two distributions, motivating a relational regularized autoencoder (RAE). In particular, besides commonly-used regularizers like the KL divergence (Dilokthanakul et al., 2016) and the Wasserstein distance (Titouan et al., 2019), which achieve *direct* comparisons of the distributions, we consider a *relational* regularizer based on the Gromov-Wasserstein (GW) distance (Mémoli, 2011) in our learning problem:

$$\min_{G,Q,p_z\in\mathcal{P}} \mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;Q}}[d(x, G(z))]$$
$$+ \gamma(\underbrace{(1-\beta)D(q_{z;Q}, p_z)}_{\text{direct comparison}} + \beta\underbrace{D_{\text{gw}}(q_{z;Q}, p_z)}_{\text{relational comparison}}). \quad (3)$$

where $\beta \in [0, 1]$ controls the trade-off between the two regularizers, and D_{gw} is the GW distance defined as follows.

Definition 2.1. *Let* $(\mathcal{X}, d_x, p_x)$ *and* $(\mathcal{Y}, d_y, p_y)$ *be two metric measure spaces, where* $(\mathcal{X}, d_x)$ *is a compact metric*

space and p_x *is a probability measure on* $\mathcal{X}$ *(with* $(\mathcal{Y}, d_y, p_y)$ *defined in the same way). The Gromov-Wasserstein distance* $D_{gw}(p_x, p_y)$ *is defined as*

$$\inf_{\pi \in \Pi(p_x, p_y)} \int_{\mathcal{X} \times \mathcal{Y}} \int_{\mathcal{X} \times \mathcal{Y}} r_{x,y,x',y'} \mathrm{d}\pi(x, y) \mathrm{d}\pi(x', y')$$

$$= \inf_{\pi \in \Pi(p_x, p_y)} \mathbb{E}_{(x,y,x',y') \sim \pi \times \pi} [r_{x,y,x',y'}]$$

where $r_{x,y,x',y'} = |d_x(x, x') - d_y(y, y')|^2$*, and* $\Pi(p_x, p_y)$ *is the set of all probability measures on* $\mathcal{X} \times \mathcal{Y}$ *with* p_x *and* p_y *as marginals.*

The $r_{x,y,x',y'}$ defines a relational loss, comparing the difference between the pairs of samples from the two distributions. Accordingly, the GW distance corresponds to the minimum expectation of the relational loss. The optimal joint distribution π^* corresponding to the GW distance is called the optimal transport between the two distributions.

The $D_{\text{gw}}(q_{z;Q}, p_z)$ in (3) penalizes the structural difference between the two distributions, mutually enhancing the clustering structure of the prior and that of the posterior. We prefer using the GW distance to implement the relational regularizer, because of the ease by which it may be combined with existing regularizers, allowing design of scalable learning algorithms. In particular, when the direct regularizer is the Wasserstein distance (Titouan et al., 2019), *i.e.*, $D = D_{\text{w}}$, we can combine it with the D_{gw} and derive a new regularizer as follows:

$$(1 - \beta) D_{\text{w}}(p_x, p_y) + \beta D_{\text{gw}}(p_x, p_y)$$
$$= (1 - \beta) \inf_{\pi \in \Pi(p_x, p_y)} \int_{\mathcal{X} \times \mathcal{Y}} c_{x,y} \mathrm{d}\pi(x, y) +$$
$$\beta \inf_{\pi \in \Pi(p_x, p_y)} \int_{\mathcal{X} \times \mathcal{Y}} \int_{\mathcal{X} \times \mathcal{Y}} r_{x,y,x',y'} \mathrm{d}\pi(x, y) \mathrm{d}\pi(x', y')$$
$$\leq \inf_{\pi \in \Pi(p_x, p_y)} \Big((1 - \beta) \underbrace{\int_{\mathcal{X} \times \mathcal{Y}} c_{x,y} \mathrm{d}\pi(x, y)}_{\text{Wasserstein term}} + \tag{4}$$
$$\beta \underbrace{\int_{\mathcal{X} \times \mathcal{Y}} \int_{\mathcal{X} \times \mathcal{Y}} r_{x,y,x',y'} \mathrm{d}\pi(x, y) \mathrm{d}\pi(x', y')}_{\text{Gromov-Wasserstein term}} \Big)$$
$$= D_{\text{fgw}}(p_x, p_y; \beta),$$

where $c : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is a direct loss function between the two spaces. The new regularizer enforces a shared optimal transport for the Wasserstein and Gromov-Wasserstein terms, corresponding to the fused Gromov-Wasserstein (FGW) distance (Vayer et al., 2018a) between the distributions. The rationality of this combination has two perspectives. First, the optimal transport indicates the correspondence between two spaces (Mémoli, 2011; Xu et al., 2019b). In the following section, we show that this optimal transport maps encoded data to the clusters defined by the prior. Enforcing shared optimal transport helps ensure the consistency of the clustering structure. Additionally, as shown in (4), $D_{\text{fgw}}(p_x, p_y; \beta) \geq (1 - \beta) D_{\text{w}}(p_x, p_y) + \beta D_{\text{gw}}(p_x, p_y)$. When replacing the regularizers in (3) with the FGW regularizer, we minimize an upper bound of the original objective function, useful from the viewpoint of optimization.

Therefore, we learn an autoencoder with relational regularization by solving the following optimization problem:

$$\min_{G, Q, p_z \in \mathcal{P}} \mathbb{E}_{p_x} \mathbb{E}_{q_{z|x;Q}} [d(x, G(z))] + \gamma D_{\text{fgw}}(q_{z;Q}, p_z; \beta), \quad (5)$$

where the prior p_z is parameterized as a Gaussian mixture model (GMM) with K components $\{\mathcal{N}(\mu_k, \Sigma_k)\}_{k=1}^{K}$. We set the probability of each component as $\frac{1}{K}$. The autoencoder can be either probabilistic or deterministic, leading to different learning algorithms.

3. Learning algorithms

3.1. Probabilistic autoencoder with hierarchical FGW

When the autoencoder is probabilistic, for each sample x, the encoder Q outputs the mean and the logarithmic variance of the posterior $q_{z|x;Q}$. Accordingly, the marginal distribution $q_{z;Q}$ becomes a GMM as well, with number of components equal to the batch size, and the regularizer corresponds to the FGW distance between two GMMs. Inspired by the hierarchical Wasserstein distance (Chen et al., 2018; Yurochkin et al., 2019; Lee et al., 2019), we leverage the structure of the GMMs and propose a hierarchical FGW distance to replace the original regularizer. In particular, given two GMMs, we define the hierarchical FGW distance between them as follows.

Definition 3.1 (Hierarchical FGW). *Let* $p = \sum_{k=1}^{K} a_k p_k$ *and* $q = \sum_{n=1}^{N} b_n q_n$ *be two GMMs.* $\{p_k\}_{k=1}^{K}$ *and* $\{q_n\}_{n=1}^{N}$ *are* M*-dimensional Gaussian distributions.* $a = [a_k] \in \Delta^{K-1}$*,* $b = [b_n] \in \Delta^{N-1}$ *are the distribution of the Gaussian components. For* $\beta \in [0, 1]$*, the hierarchical fused Gromov-Wasserstein distance between these two GMMs is*

$$D_{hfgw}(p, q; \beta)$$
$$= \min_{T=[t_{kn}] \in \Pi(a,b)} (1 - \beta) \sum_{k,n} D_w(p_k, q_n) t_{kn} +$$
$$\beta \sum_{k,k',n,n'} |D_w(p_k, p_{k'}) - D_w(q_n, q_{n'})|^2 t_{kn} t_{k'n'} \tag{6}$$
$$= D_{fgw}(a, b; \beta).$$

As shown in (6), the hierarchical FGW corresponds to an FGW distance between the distributions of the Gaussian components, whose ground distance is the Wasserstein distance between the Gassuain components. Figures 2(a) and 2(b) further illustrate the difference between the FGW and our hierarchical FGW. For the two GMMs, instead of computing the optimal transport between them in the sample space, the hierarchical FGW builds optimal transport between their Gaussian components. Additionally, we have

Proposition 3.2. $D_{hfgw}(p, q; \beta) = D_{fgw}(p, q)$ *when* $\Sigma = 0$ *for all the Gaussian components.*

Replacing the FGW with the hierarchical FGW, we convert an optimization problem of a continuous distribution (the π
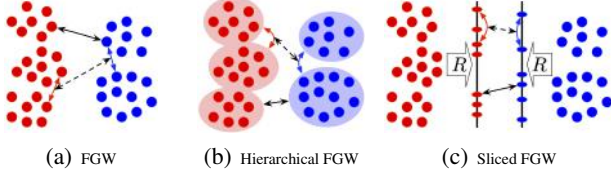
Figure 2. Illustrations of FGW, hierarchical FGW, and sliced FGW. The black solid arrow represents the distance between samples while the black dotted arrow represents the relational loss between sample pairs. In (a, c), the red and blue arrows represent the Euclidean distance between samples. In (b), the red and blue arrows represent the Wasserstein distance between Gaussian components.

Algorithm 1 Learning RAE with hierarchical FGW

1: **Input** Samples in $\mathcal{X}$
2: **Output** The autoencoder $\{G, Q\}$ and the prior with K Gaussian components $p_z = \frac{1}{K}\sum_k \mathcal{N}(z; \mu_k, \text{diag}(\sigma_k))$.
3: **for** each epoch
4:     **for** each batch of samples $\{x_n\}_{n=1}^N \subset \mathcal{X}$
5:         $\mu_n, \log(\sigma_n^2) = Q(x_n)$ for $n = 1, ..., N$.
6:         Reparameterize $z_n = \mu_n + \epsilon\sigma_n$, where $\epsilon \sim \mathcal{N}(0, I)$.
7:         $q_{z;Q} = \frac{1}{N}\sum_n \mathcal{N}(z; \mu_n, \text{diag}(\sigma_n))$.
8:         Calculate D_p, D_q via (10), calculate D via (8)
9:         Obtain optimal transport T^* via solving (7).
10:        $L_{\text{reconstruction}} = \sum_n d(x_n, G(z_n))$.
11:        $D_{\text{hfgw}}(q_{z;Q}, p_z; \beta) = \langle D - 2\beta D_p T^* D_q^\top, T^*\rangle$.
12:        Update $G, Q, p_z = \text{Adam}(L_{\text{reconstruction}} + \gamma D_{\text{hfgw}})$.

---

in (4)) to a much simpler optimization problem of a discrete distribution (the T in (6)). Rewriting (6) in matrix form, we compute the hierarchical FGW distance via solving the following non-convex optimization problem:

$$D_{\text{hfgw}}(p, q; \beta) = \min_{T \in \Pi(a,b)} \langle D - 2\beta D_p T D_q^\top, T\rangle, \quad (7)$$

where $\langle \cdot, \cdot \rangle$ indicates the inner product between matrices, $\Pi(a, b) = \{T \geq 0 | T1_N = a, T^\top 1_K = b\}$, and 1_N is an N-dimensional all-one vector. The optimal transport matrix T is a joint distribution of the Gaussian components in the two GMMs, where $D_p = [D_\text{w}(p_k, p_{k'})] \in \mathbb{R}^{K \times K}$ and $D_q = [D_\text{w}(q_n, q_{n'})] \in \mathbb{R}^{N \times N}$, whose elements are the Wasserstein distances between Gaussian components, and

$$D = (1 - \beta)D_{pq} + \frac{\beta}{K}(D_p \odot D_p) + \frac{\beta}{N}(D_q \odot D_q)^\top \quad (8)$$

with $D_{pq} = [D_\text{w}(p_k, q_n)] \in \mathbb{R}^{K \times N}$ and $\odot$ represents the Hadamard product. The Wasserstein distance between Gaussian distributions has a closed form:

Definition 3.3. *Let $p = \mathcal{N}(u_p, \Sigma_p)$ and $q = \mathcal{N}(u_q, \Sigma_q)$ be two N-dimensional Gaussian distributions, where u and Σ represent the mean and the covariance matrix, respectively. The Wasserstein distance $D_w(p, q)$ is*

$$\|u_p - u_q\|_2^2 + trace(\Sigma_p + \Sigma_q - 2(\Sigma_p^{\frac{1}{2}}\Sigma_q\Sigma_p^{\frac{1}{2}})^{\frac{1}{2}}). \quad (9)$$

When the covariance matrices are diagonal, *i.e.*, $\Sigma = \text{diag}(\sigma^2)$, where $\sigma = [\sigma_n] \in \mathbb{R}^N$ is the standard deviation, (9) can be rewritten as

$$D_\text{w}(p, q) = \|u_p - u_q\|_2^2 + \|\sigma_p - \sigma_q\|_2^2. \quad (10)$$

We solve (7) via the proximal gradient method in (Xu et al., 2019b), with further details in the Supplementary Material.

The hierarchical FGW is a good substitute for the original FGW, imposing structural constraints while being more efficient computationally. Plugging the hierarchical FGW and its computation into (5), we apply Algorithm 1 to learn the proposed RAE. Note that taking advantage of the Envelope

Theorem (Afriat, 1971), we treat the optimal transport matrix as constant when applying backpropagation, reducing computational complexity significantly. The optimal transport matrix maps the components in the $q_{z;Q}$ to those in the p_z. Because the components in the $q_{z;Q}$ correspond to samples and the components in the p_z correspond to clusters, this matrix indicates the clustering structure of the samples.

3.2. Deterministic autoencoder with sliced FGW

When the autoencoder is deterministic, its encoder outputs the latent codes corresponding to observed samples. These latent codes can be viewed as the samples of $q_{z;Q}$. For the prior p_z, we can also generate samples with the help of the reparameterization trick. In such a situation, we estimate the FGW distance in (5) based on the samples of the two distributions. For arbitrary two metric measure spaces $(\mathcal{X}, d_x, p_x)$ and $(\mathcal{Y}, d_y, p_y)$, the empirical FGW between their samples $\{x_i\}_{i=1}^N$ and $\{y_j\}_{j=1}^N$ is

$$\widehat{D}_{\text{fgw}}(p_x, p_y; \beta)$$
$$= \min_{T \in \Pi(\frac{1}{N}1_N, \frac{1}{N}1_N)}(1 - \beta)\sum_{i,j=1}^N d(x_i, y_j)t_{ij} + \quad (11)$$
$$\beta \sum_{i,i',j,j'} |d(x_i, x_{i'}) - d(y_j, y_{j'})|^2 t_{ij} t_{i'j'}.$$

We can rewrite this empirical FGW in matrix form as (7), and solve it by the proximal gradient method discussed above. When the samples are in 1D space and the metric is the Euclidean distance, however, according to the sliced GW distance in (Titouan et al., 2019) and the sliced Wasserstein distance in (Kolouri et al., 2016), the optimal transport matrix corresponds to a permutation matrix and the $\widehat{D}_{\text{fgw}}(p_x, p_y; \beta)$ can be rewritten as:

$$\widehat{D}_{\text{fgw}}(p_x, p_y; \beta) = \min_{\sigma \in \mathcal{P}_N} \frac{1 - \beta}{N}\sum_{i=1}^N (x_i - y_{\sigma(i)})^2 +$$
$$\frac{\beta}{N}\sum_{i,j=1}^N ((x_i - x_j)^2 - (y_{\sigma(i)} - y_{\sigma(j)})^2)^2, \quad (12)$$

where $\mathcal{P}_N$ is the set of all permutations of $\{1, ..., N\}$. Without loss of generality, we assume the 1D samples are sorted,

i.e., $x_1 \leq ... \leq x_N$ and $y_1 \leq ... \leq y_N$, and demonstrate that the solution of (12) is characterized by the following theorem.

Theorem 3.4. *For $x, y \in \{x = [x_i], y = [y_j] \in \mathbb{R}^N \times \mathbb{R}^N | x_1 \leq ... \leq x_N, y_1 \leq ... \leq y_N\}$, we denote their zero-mean translations as x' and y', respectively. The solution of (12) satisfies: 1) When $(\sum_i x'_i y'_i + \frac{1-\beta}{8\beta})^2 \geq (\sum_i x'_i y'_{n+1-i} + \frac{1-\beta}{8\beta})^2$, the solution is the identity permutation $\sigma(i) = i$. 2) Otherwise, the solution is the anti-identity permutation $\sigma(i) = n + 1 - i$.*

The proof of Theorem 3.4 is provided in the Supplementary Material. Consequently, for the samples in 1D space, we can calculate the empirical FGW distance via permuting the samples. To leverage this property for high-dimensional samples, we propose the following sliced FGW distance:

Definition 3.5 (Sliced FGW). *Let $\mathcal{S}^{M-1} = \{\theta \in \mathbb{R}^M | \|\theta\|_2 = 1\}$ be the M-dimensional hypersphere and $u_{\mathcal{S}^{M-1}}$ the uniform measure on $\mathcal{S}^{M-1}$. For each θ, we denote the projection on θ as R_θ, where $R_\theta(x) = \langle x, \theta \rangle$. For $(\mathcal{X}, d_x, p_x)$ and $(\mathcal{Y}, d_y, p_y)$, we define their sliced fused Gromov-Wasserstein distance as*

$$D_{sfgw}(p_x, p_y; \beta) = \mathbb{E}_{\theta \sim u_{\mathcal{S}^{M-1}}}[D_{fgw}(R_\theta \# p_x, R_\theta \# p_y; \beta)]],$$

where $R_\theta \# p$ represents the distribution after the projection, and $D_{fgw}(R_\theta \# p_x, R_\theta \# p_y)$ is the FGW distance between $(R_\theta(\mathcal{X}), d_{R_\theta(x)}, R_\theta \# p_x)$ and $(R_\theta(\mathcal{Y}), d_{R_\theta(y)}, R_\theta \# p_y)$.

According to this definition, the sliced FGW projects the original metric measure spaces into 1D spaces, and calculates the FGW distance between these spaces. The sliced FGW corresponds to the expectation of the FGW distances under different projections. We can approximate the sliced FGW distance based on the samples of the distributions as well. In particular, given $\{x_i\}_{i=1}^N$ from $\mathcal{X}$, $\{y_i\}_{i=1}^N$ from $\mathcal{Y}$, and L projections $\{R_{\theta_l}\}_{l=1}^L$, the empirical sliced FGW is

$$\widehat{D}_{sfgw}(p_x, p_y; \beta)$$
$$= \frac{1}{L} \sum_{l=1}^L \widehat{D}_{fgw}(R_{\theta_l} \# p_x, R_{\theta_l} \# p_y; \beta)$$
$$= \frac{1}{L} \sum_{l=1}^L \min_{\sigma \in \mathcal{P}_N} \frac{1-\beta}{N} \sum_{i=1}^N (x_{i,\theta_l} - y_{\sigma(i),\theta_l})^2 + \quad (13)$$
$$\frac{\beta}{N} \sum_{i,j=1}^N ((x_{i,\theta_l} - x_{j,\theta_l})^2 - (y_{\sigma(i),\theta_l} - y_{\sigma(j),\theta_l})^2)^2,$$

where $x_{i,\theta_l} = R_{\theta_l}(x_i)$ represents the projected sample. Figure 2(c) further illustrates the principle of the sliced FGW distance. Replacing the empirical FGW with the empirical sliced FGW, we learn the relational regularized autoencoder via Algorithm 2.

3.3. Comparisons on computational complexity

Compared with calculating empirical FGW distance directly, our hierarchical FGW and sliced FGW have much lower

---

Algorithm 2 Learning RAE with sliced FGW

1: **Input** Samples in $\mathcal{X}$
2: **Output** The autoencoder $\{G, Q\}$ and the prior with K Gaussian components $p_z = \frac{1}{K} \sum_k \mathcal{N}(z; \mu_k, \text{diag}(\sigma_k))$.
3: **for** each epoch
4:   **for** each batch of samples $\{x_n\}_{n=1}^N \subset \mathcal{X}$
5:     **for** $n = 1, ..., N$
6:       Samples of $q_{z;Q}$: $z_n = Q(x_n)$.
7:       Samples of p_z: $k \sim \text{Categorical}(K)$, $z'_n = \mu_k + \epsilon\sigma_k$.
8:     **for** $l = 1, ..., L$
9:       Create a random projection $\theta_l \in \mathcal{S}^{M-1}$
10:       $z_{n,\theta_l} = R_{\theta_l} z_n$, $z'_{n,\theta_l} = R_{\theta_l} z'_n$ for $n = 1, ..., N$.
11:       Sort $\{z_{n,\theta_l}\}_{n=1}^N$ and $\{z'_{n,\theta_l}\}_{n=1}^N$, respectively.
12:       Calculate $\widehat{D}_{fgw}(R_{\theta_l} \# q_{z;Q}, R_{\theta_l} \# p_z; \beta])$ based on sorted samples and Theorem 3.4.
13:     $L_{reconstruction} = \sum_n d(x_n, G(z_n))$.
14:     Calculate $\widehat{D}_{sfgw}(q_{z;Q}, p_z; \beta)$ via (13).
15:     Update $G, Q, p_z = \text{Adam}(L_{reconstruction} + \gamma\widehat{D}_{sfgw})$.

---

computational complexity. Following notation in the previous two subsections, we denote the batch size as N, the number of Gaussian components in the prior as K, and the dimension of the latent code as M. If we apply the proximal gradient method in (Xu et al., 2019b) to calculate the empirical FGW directly, the computational complexity is $\mathcal{O}(JN^3)$, where J is the number of Sinkhorn iterations used in the algorithm. For our hierarchical FGW, we apply the proximal gradient method to a problem with a much smaller size (i.e., solving (7)) because of $K \ll N$ in general. Accordingly, the computational complexity becomes $\mathcal{O}(JN^2K)$. For our sliced FGW, we apply L random projections to project the latent codes to 1D spaces, whose complexity is $\mathcal{O}(LMN)$. For each pair of projected samples, we sort them with $\mathcal{O}(N \log N)$ operations and compute (12) with $O(N^2)$ operations. Overall, the computational complexity of our sliced FGW is $\mathcal{O}(LN(M + \log N + N))$. Because $J \approx L$ in general, the computational complexity of the sliced FGW is comparable to that of the hierarchical FGW.

4. Relational Co-Training of Autoencoders

Besides learning a single autoencoder, we can apply our relational regularization to learn multiple autoencoders. As shown in Figure 1(b), when learning two autoencoders we can penalize the GW distance between their posterior distributions, and accordingly the learning problem becomes:

$$\min_{\{G_s, Q_s\}_{s=1}^2} \sum_{s=1}^2 \Big(\mathbb{E}_{p_{x_s}} \mathbb{E}_{q_{z_s|x_s;Q_s}}[d(x_s, G_s(z_s))] + \\ \gamma(1-\tau)D(q_{z_s;Q_s}, p_{z_s}) \Big) + 2\gamma\tau D_{gw}(q_{z_1;Q_1}, q_{z_2;Q_2}). \quad (14)$$

The regularizer D quantifies the discrepancy between the marginalized posterior and the prior, the prior distributions can be predefined or learnable parameters, $\tau \in [0, 1]$ achieves a trade-off between D and the relational regularizer

D_{gw}, and γ controls the overall significance of these two kinds of regularizers. When learning probabilistic autoencoders, we set D to the hierarchical Wasserstein distance between GMMs (Chen et al., 2018) and approximate the relational regularizer by a hierarchical GW distance, equivalent to the hierarchical FGW with $\beta = 1$. When learning deterministic autoencoders, we set D to the sliced Wasserstein distance used in (Kolouri et al., 2018) and approximate the relational regularizer via the sliced GW (Titouan et al., 2019) (the sliced FGW with $\beta = 1$).

The main advantage of the proposed relational regularization is that it is applicable for co-training heterogeneous autoencoders. As shown in (14), the data used to train the autoencoders can come from different domains and with different data distributions. To fully capture the information in each domain, sometimes the autoencoders have heterogeneous architectures, and the corresponding latent codes are in incomparable spaces, *e.g.*, with different dimensions. Taking the GW distance as the relational regularizer, we impose a constraint on the posterior distributions defined in different latent spaces, encouraging structural similarity between them. This regularizer helps avoid over-regularization because it does not enforce a shared latent distribution across different domains. Moreover, the proposed regularizer is imposed on the posterior distributions. In other words, it does not require samples from different domains to be paired.

According to the analysis above, our relational co-training strategy has potential for multi-view learning, especially in the scenario with unpaired samples. In particular, given the data in different domains, we first learn their latent codes via solving (14). Concatenating the latent codes in different domains, we can use the concatenation of the latent codes as the features for downstream learning tasks.

5. Related Work

Gromov-Wasserstein distance The GW distance has been used as a metric for shape registration (Mémoli, 2009; 2011), vocabulary set alignment (Alvarez-Melis & Jaakkola, 2018), and graph matching (Chowdhury & Mémoli, 2018; Vayer et al., 2018b; Xu et al., 2019b). The work in (Peyré et al., 2016) proposes an entropy-regularized GW distance and calculates it based on Sinkhorn iterations (Cuturi, 2013). Following this direction, the work in (Xu et al., 2019b) replaces the entropy regularizer with a Bregman proximal term. The work in (Xu, 2019) proposes an ADMM-based method to calculate the GW distance. To further reduce the computational complexity, the recursive GW distance (Xu et al., 2019a) and the sliced GW distance (Titouan et al., 2019) have been proposed. For generative models, the work in (Bunne et al., 2019) leverages the GW distance to learn coupled adversarial generative networks. However, none of the existing autoencoders consider using the GW distance

Table 1. Comparisons for different autoencoders

Method	$Q : \mathcal{X} \mapsto \mathcal{Z}$	p_z	$D(q_{z;Q}, p_z)$
VAE	Probabilistic	$\mathcal{N}(z; 0, I)$	KL
WAE	Deterministic	$\mathcal{N}(z; 0, I)$	MMD
SWAE	Deterministic	$\mathcal{N}(z; 0, I)$	D_{w}
GMVAE	Probabilistic	$\frac{1}{K} \sum_k \mathcal{N}(z; u_k, \Sigma_k)$	KL
VampPrior	Probabilistic	$\frac{1}{K} \sum_k \mathcal{N}(z; Q(x_k))$	KL
Our RAE	Probabilistic	$\frac{1}{K} \sum_k \mathcal{N}(z; u_k, \Sigma_k)$	D_{hfgw}
	Deterministic		$\widehat{D}_{\text{sfgw}}$

as their regularizer.

Autoencoders The principle of the autoencoder is to minimize the discrepancy between the data and model distributions. The common choices of the discrepancy include the KL divergence (Kingma & Welling, 2013; Dilokthanakul et al., 2016; Tomczak & Welling, 2018; Takahashi et al., 2019) and the Wasserstein distance (Tolstikhin et al., 2018; Kolouri et al., 2018), which lead to different learning algorithms. Our relational regularized autoencoder can be viewed as a new member of the Wasserstein autoencoder family. Compared with the MMD and the GAN loss used in WAE (Tolstikhin et al., 2018), and the sliced Wasserstein distance used in (Kolouri et al., 2018), our FGW-based regularizer imposes relational constraints and allows the learning of an autoencoder with structured prior distribution.

Co-training methods For data in different domains, a commonly-used co-training strategy maps them to a shared latent space, and encourages similarity between their latent codes. This co-training strategy suppresses the risk of overfitting for each model and enhances their generalization power, which achieves encouraging performance in multi-view learning (Kumar & Daumé, 2011; Chen & Denoyer, 2017; Sindhwani et al., 2005). However, this strategy assumes that the latent codes yield the same distribution, which may lead to over regularization. Additionally, it often requires well-aligned data, *i.e.*, the samples in different domains are paired. Our relational co-training strategy provides a potential solution to relax these restrictions for practical applications.

6. Experiments

6.1. Image generation

We test our relational regularized autoencoder (**RAE**) for image-generation tasks and compare it with the following alternatives: the variational autoencoder (VAE) (Kingma & Welling, 2013), the Wasserstein autoencoder (WAE) (Tolstikhin et al., 2018), the sliced Wasserstein autoencoder (SWAE) (Kolouri et al., 2018), the Gaussian mixture VAE (GMVAE) (Dilokthanakul et al., 2016), and the VampPrior (Tomczak & Welling, 2018). Table 1 lists the main differences between our RAE and these baselines.

Table 2. Comparisons on learning image generator

Encoder $Q : \mathcal{X} \mapsto \mathcal{Z}$	Method	MNIST		CelebA	
		Rec. loss	FID	Rec. loss	FID
Probabilistic	VAE	16.60	156.11	96.36	59.99
	GMVAE	16.76	60.88	108.13	353.17
	VampPrior	22.41	127.81	—	—
	RAE	14.14	41.99	63.21	52.20
Deterministic	WAE	9.97	52.78	63.83	52.07
	SWAE	11.10	35.63	87.02	88.91
	RAE	10.37	49.39	64.49	51.45

Table 3. Runtime per epoch (second) when training various models

Dataset	WAE	SWAE	P-RAE	D-RAE
MNIST	25.6	24.7	26.9	24.8
CelebA	602.2	553.4	618.5	569.7

We test the methods on the MNIST (LeCun et al., 1998) and CelebA datasets (Liu et al., 2015). For fairness, all the autoencoders have the same DCGAN-style architecture (Radford et al., 2015) and are learned with the same hyperparameters: the learning rate is 0.001; the optimizer is Adam (Kingma & Ba, 2014) with $\beta_1 = 0.5$ and $\beta_2 = 0.999$; the number of epochs is 50; the batch size is 100; the weight of regularizer γ is 1; the dimension of latent code is 8 for MNIST and 64 for CelebA. For the autoencoders with structured priors, we set the number of the Gaussian components to be 10 and initialize their prior distributions at random. For the proposed RAE, the hyperparameter β is set to be 0.1, which empirically makes the Wasserstein term and the GW term in our FGW distance have the same magnitude. The probabilistic RAE calculates the hierarchical FGW based on the proximal gradient method with 20 iterations, and the deterministic RAE calculates the sliced FGW with 50 random projections. All the autoencoders use Euclidean distance as the distance between samples, thus the reconstruction loss is the mean-square-error (MSE). We implement all the autoencoders with PyTorch and train them on a single NVIDIA GTX 1080 Ti GPU. More implementation details, $e.g.$, the architecture of the autoencoders, are provided in Supplementary Material.

For each dataset, we compare the proposed RAE with the baselines on i) the reconstruction loss on testing samples; ii) the Fréchet Inception Distance (FID) between 10,000 testing samples and 10,000 randomly generated samples. We list the performance of various autoencoders in Table 2. Among probabilistic autoencoders, our RAE consistently achieves the best performance on both testing reconstruction loss and FID score. When learning deterministic autoencoders, our RAE is at least comparable to the considered alternatives on these measurements. Figure 3 compares the autoencoders on their convergence of the reconstruction loss. The convergence of our RAE is almost the same as that of state-of-the-art methods, which further verifies its feasibility.

For the autoencoders learning GMMs as their priors, we further make comparisons for them in conditional generation tasks, $i.e.$, generating samples conditioned on specific Gaussian components. Figures 4 and 5 visualize the generation results for various methods. For the MNIST dataset, the GMVAE, our probabilistic RAE, and deterministic RAE achieve desired generation results. The images conditioned on different Gaussian components correspond to different digits/writing styles. The VampPrior, however, suffers from a problem of severe mode collapse. The images conditioned on different Gaussian components are similar to each other and with limited modes – most of them are "0", "2", "3", and "8". As shown in Table 1, for each Gaussian component of the prior, the VampPrior parameterizes it by passing a landmark x_k through the encoder. Because the landmarks are in the sample space, this implicit model requires more parameters, making it sensitive to initialization with a high risk of overfitting. Figures 3(a) and 3(b) verify our claim: the testing loss of the VampPrior is unstable and does not converge well during training. For the CelebA dataset, the GMVAE fails to learn a GMM-based prior. As shown in Figure 5(a), the GMVAE trains a single Gaussian distribution, while ignoring the remaining components. As a result, only one Gaussian component can generate face images. Our probabilistic and deterministic RAE, by contrast, learn their GMM-based prior successfully. In particular, all the components of our probabilistic RAE can generate face images, but the components are indistinguishable. Our deterministic RAE achieves the best performance in this conditional generation task – different components can generate semantically-meaningful images with interpretable diversity. For each component, we add some tags to highlight semantic meaning. The visual comparisons for various autoencoders on their reconstructed and generated samples are shown in Supplementary Material.

6.2. Multi-view learning via co-training autoencoders

We test our relational co-training strategy on four multi-view learning datasets (Li et al., 2015):[1] **Caltech101-7** is a subset of the Caltech-101 dataset (Fei-Fei et al., 2004) with 1,474 images in 7 classes. Each image is represented by 48-dimensional Gabor features and 40-dimensional Wavelet moments. **Caltech101-20** is a subset of the Caltech-101 with 2,386 images in 20 classes. The features are the same with the Caltech101-7. **Handwritten** is a dataset with 2,000 images corresponding to 10 digits. Each image has 240-dimensional pixel-based features and 76-dimensional Fourier coefficients. **Cathgen** is a real-world dataset of 8,000 patients. For each patient, we seek to leverage 44-dimensional clinical features and 67-dimensional genetic
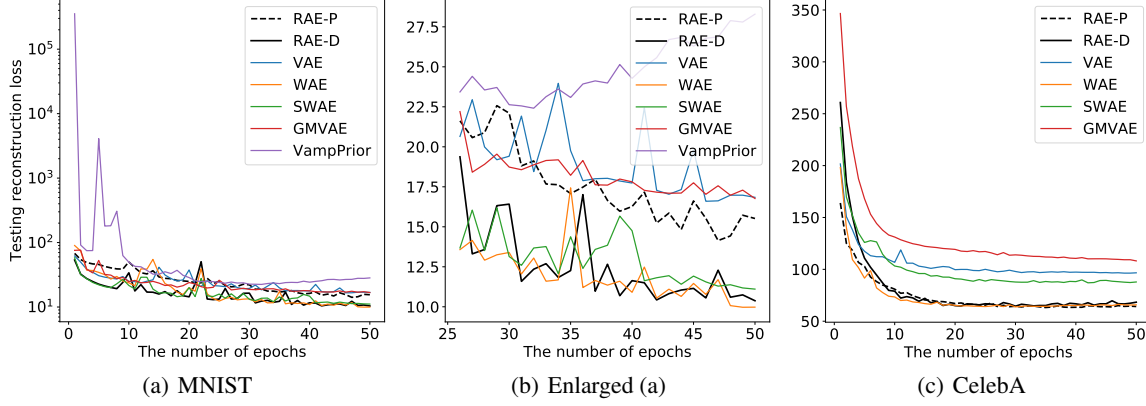
---

[1] https://github.com/yeqinglee/mvdata

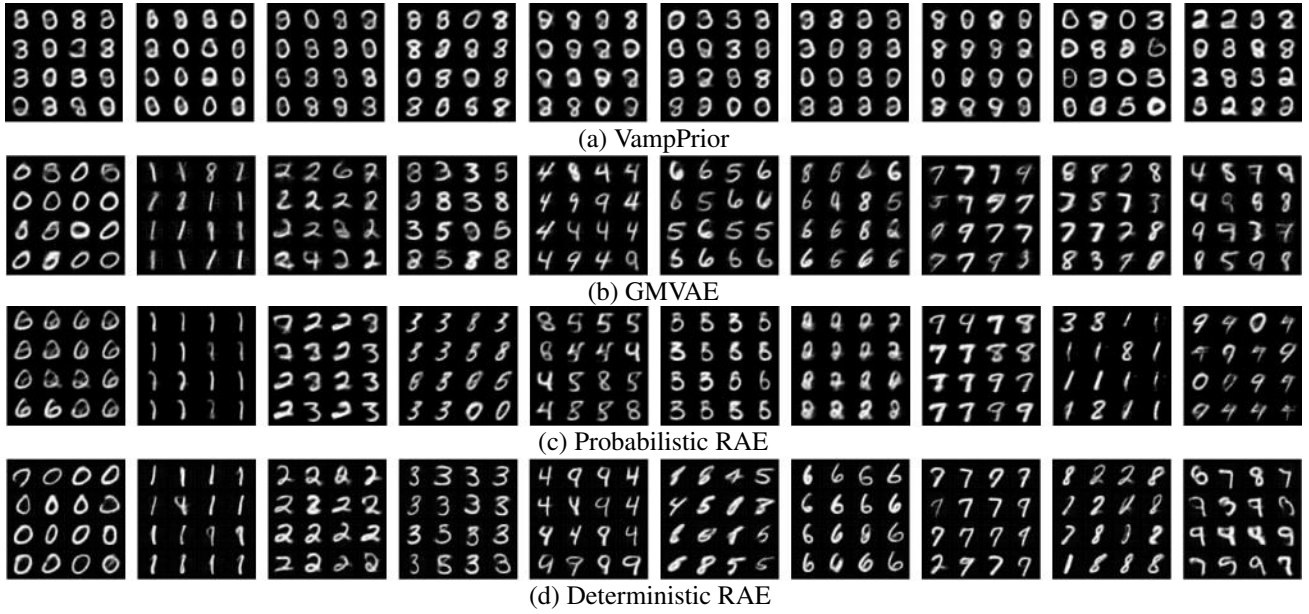*Figure 3.* Comparisons for various methods on their convergence.

Figure 4. Comparisons on conditional digit generation.

features to predict the happening of myocardial infarction, *i.e.*, a binary classification task.

For each dataset, we use 80% of the data for training, 10% for validation, and the remaining 10% for testing. We test various multi-view learning methods. For each method, we first learn two autoencoders for the data in different views in an unsupervised way, and then concatenate the latent codes of the autoencoders as the features and train a classifier based on softmax regression. When learning autoencoders, our relational co-training method solves (14) with $\gamma = 1$ and $\tau = 0.5$. The influence of τ on the learning results is shown in Supplementary Material. For simplification, we set the prior distributions as normal distributions in (14). The autoencoders are probabilistic, whose encoders and decoders are MLPs. Each autoencoder has 20-dimensional latent codes, and more implementation details are provided in Supplementary Material. We set D as the hierarchical

Wasserstein distance and the relational regularizer as the hierarchical GW distance. In addition to the proposed method, denoted as **AEs+GW**, we consider the following baselines: *i*) learning two variational autoencoders independently (**Independent AEs**); *ii*) learning two variational autoencoders jointly with a least-square co-regularization (Sindhwani et al., 2005) (**AEs+CoReg**); *iii*) learning latent representations via canonical correlation analysis (**CCA**) (Vía et al., 2007); *iv*) learning two autoencoders jointly with a CCA-based regualrization (**AEs+CCA**) (Wang et al., 2015); *v*) learning two autoencoders by replacing the D_{gw} in (14) with a Wasserstein regularizer (**AEs+W**). The AE+CoReg penalizes the Euclidean distance between the latent codes from different views, which needs paired samples. The remaining methods penalize the discrepancy between the distributions of the latent codes, which are applicable for unpaired samples. The classification accuracy in Table 4
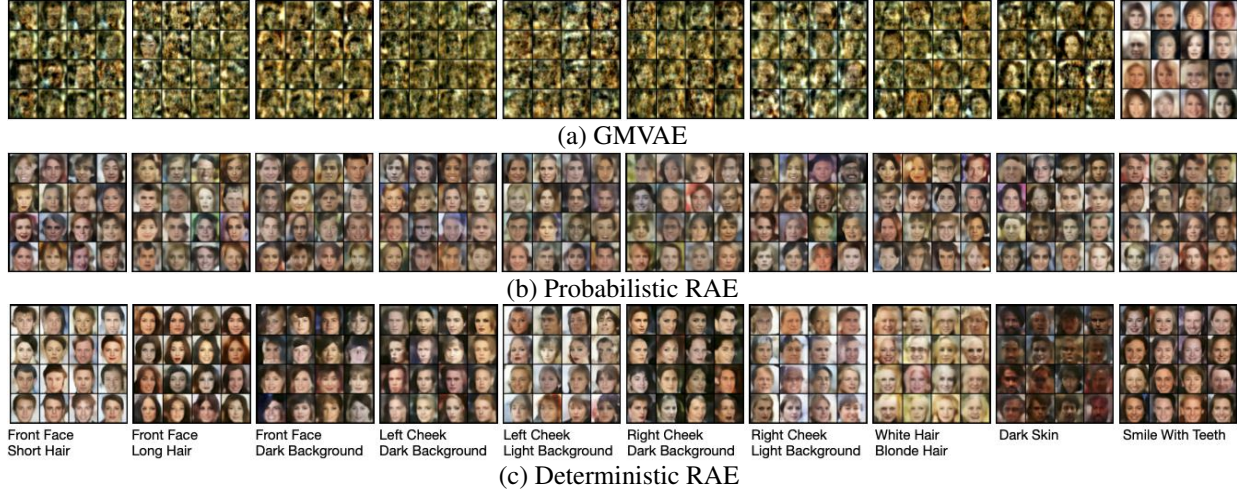
(a) GMVAE

(b) Probabilistic RAE

| Front Face | Front Face | Front Face | Left Cheek | Left Cheek | Right Cheek | Right Cheek | White Hair | Dark Skin | Smile With Teeth |
| Short Hair | Long Hair | Dark Background | Dark Background | Light Background | Dark Background | Light Background | Blonde Hair | | |

(c) Deterministic RAE

Figure 5. Comparisons on conditional face generation.

Table 4. Comparisons on classification accuracy (%)

Method	Data type	Caltech101-7	Caltech101-20	Handwritten	Cathgen
Independent AEs	Unpaired	56.92±1.67	33.07±2.09	52.09±5.82	64.36±1.93
AEs+CoReg (Sindhwani et al., 2005)	Paired	76.58±1.38	60.25±1.66	56.20±5.25	66.79±1.30
CCA (Vía et al., 2007)	Paired	78.33±1.88	52.27±2.32	66.28±5.02	65.28±2.17
AEs+CCA (Wang et al., 2015)	Paired	80.24±1.22	62.37±1.35	69.72±4.64	66.89±1.57
AEs+W	Unpaired	83.07±1.69	**69.58**±2.03	71.21±5.55	66.06±1.68
AEs+GW (Ours)	Unpaired	**84.29**±1.74	69.39±2.01	**72.36**±3.82	**66.99**±1.77

demonstrates effectiveness of our relational co-training strategy, as the proposed method outperforms the baselines consistently across different datasets.

7. Conclusions

A new framework has been proposed for learning autoencoders with relational regularization. Leveraging the GW distance, this framework allows the learning of structured prior distributions associated with the autoencoders and prevents the model from under-regularization. Besides learning a single autoencoder, the proposed relational regularizer is beneficial for co-training heterogeneous autoencoders. In the future, we plan to make this relational regularizer applicable for co-training more than two autoencoders and further reduce its computational complexity.

References

Afriat, S. Theory of maxima and the method of lagrange. *SIAM Journal on Applied Mathematics*, 20(3):343–357, 1971.

Alvarez-Melis, D. and Jaakkola, T. Gromov-wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1881–1890, 2018.

Bowman, S., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 10–21, 2016.

Bunne, C., Alvarez-Melis, D., Krause, A., and Jegelka, S. Learning generative models across incomparable spaces. In *International Conference on Machine Learning*, pp. 851–861, 2019.

Chen, M. and Denoyer, L. Multi-view generative adversarial networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 175–188. Springer, 2017.

Chen, Y., Georgiou, T. T., and Tannenbaum, A. Optimal transport for gaussian mixture models. *IEEE Access*, 7: 6269–6278, 2018.

Chowdhury, S. and Mémoli, F. The Gromov-Wasserstein distance between networks and stable network invariants. *arXiv preprint arXiv:1808.04337*, 2018.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pp. 2292–2300, 2013.

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.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR workshop*, pp. 178–178. IEEE, 2004.

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

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kolouri, S., Zou, Y., and Rohde, G. K. Sliced Wasserstein kernels for probability distributions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5258–5267, 2016.

Kolouri, S., Pope, P. E., Martin, C. E., and Rohde, G. K. Sliced-wasserstein autoencoder: an embarrassingly simple generative model. *arXiv preprint arXiv:1804.01947*, 2018.

Kumar, A. and Daumé, H. A co-training approach for multi-view spectral clustering. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 393–400, 2011.

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

Lee, J., Dabagia, M., Dyer, E., and Rozell, C. Hierarchical optimal transport for multimodal distribution alignment. In *Advances in Neural Information Processing Systems*, pp. 13453–13463, 2019.

Li, Y., Nie, F., Huang, H., and Huang, J. Large-scale multi-view spectral clustering via bipartite graph. In *AAAI*, 2015.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *ICCV*, 2015.

Mémoli, F. Spectral Gromov-Wasserstein distances for shape matching. In *ICCV Workshops*, pp. 256–263, 2009.

Mémoli, F. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.

Peyré, G., Cuturi, M., and Solomon, J. Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*, pp. 2664–2672, 2016.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Sindhwani, V., Niyogi, P., and Belkin, M. A co-regularization approach to semi-supervised learning with multiple views. In *Proceedings of ICML workshop on learning with multiple views*, volume 2005, pp. 74–79, 2005.

Takahashi, H., Iwata, T., Yamanaka, Y., Yamada, M., and Yagi, S. Variational autoencoder with implicit optimal priors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5066–5073, 2019.

Titouan, V., Flamary, R., Courty, N., Tavenard, R., and Chapel, L. Sliced gromov-wasserstein. In *Advances in Neural Information Processing Systems*, pp. 14726–14736, 2019.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schölkopf, B. Wasserstein auto-encoders. In *International Conference on Learning Representations (ICLR 2018)*. OpenReview. net, 2018.

Tomczak, J. and Welling, M. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223, 2018.

Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. Fused Gromov-Wasserstein distance for structured objects: theoretical foundations and mathematical properties. *arXiv preprint arXiv:1811.02834*, 2018a.

Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. Optimal transport for structured data. *arXiv preprint arXiv:1805.09114*, 2018b.

Vía, J., Santamaría, I., and Pérez, J. A learning algorithm for adaptive canonical correlation analysis of several data sets. *Neural Networks*, 20(1):139–152, 2007.

Wang, S., Ding, Z., and Fu, Y. Coupled marginalized autoencoders for cross-domain multi-view learning. In *IJCAI*, pp. 2125–2131, 2016.

Wang, W., Arora, R., Livescu, K., and Bilmes, J. On deep multi-view representation learning. In *International Conference on Machine Learning*, pp. 1083–1092, 2015.

Wang, W., Gan, Z., Xu, H., Zhang, R., Wang, G., Shen, D., Chen, C., and Carin, L. Topic-guided variational auto-encoder for text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 166–177, 2019.

Xu, H. Gromov-wasserstein factorization models for graph clustering. *arXiv preprint arXiv:1911.08530*, 2019.

Xu, H., Luo, D., and Carin, L. Scalable gromov-wasserstein learning for graph partitioning and matching. *arXiv preprint arXiv:1905.07645*, 2019a.

Xu, H., Luo, D., Zha, H., and Duke, L. C. Gromov-wasserstein learning for graph matching and node embedding. In *International Conference on Machine Learning*, pp. 6932–6941, 2019b.

Ye, T., Wang, T., McGuinness, K., Guo, Y., and Gurrin, C. Learning multiple views with orthogonal denoising autoencoders. In *International Conference on Multimedia Modeling*, pp. 313–324. Springer, 2016.

Yurochkin, M., Claici, S., Chien, E., Mirzazadeh, F., and Solomon, J. Hierarchical optimal transport for document representation. *arXiv preprint arXiv:1906.10827*, 2019.

A. The proximal gradient method

Both the hierarchical FGW in (6) and the empirical FGW in (11) can be rewritten in matrix format. As shown in (7), the calculation of the distance corresponds to solving the following non-convex optimization problem:

$$\min_{T \in \Pi(p,q)} \langle D - 2D_p T D_q^\top, T \rangle, \tag{15}$$

where p and q are predefined discrete distributions. This problem can be solved iteratively by the following proximal gradient method (Xu et al., 2019b). In each j-th iteration, given current estimation $T^{(j)}$, we consider the following problem with a proximal term

$$\min_{T \in \Pi(p,q)} \langle D - 2D_p T^{(j)} D_q^\top, T \rangle + \alpha \mathrm{KL}(T \| T^{(j)}). \tag{16}$$

This subproblem can be solved easily via Sinkhorn iterations (Cuturi, 2013). The details of the algorithm are shown in Algorithm 3. In our experiments, we set $J = 20$ when learning probabilistic RAE (P-RAE). The hyperparameter α is set adaptively. In particular, in each iteration, given the matrix $C^{(j)} = D - 2D_p T^{(j)} D_q^\top$, we set $\alpha = 0.1 \max(C^{(j)})$. This setting helps us improve the numerical stability when calculating the Φ in Algorithm 3. When learning deterministic RAE (D-RAE), we apply the sliced FGW distance with $L = 50$ random projections, such that the training time of the D-RAE and that of the P-RAE are comparable.

Algorithm 3 $\min_{T \in \Pi(p,q)} \langle D - 2D_p T D_q^\top, T \rangle$

1: Initialize $T^{(0)} = pq^\top$, $a = p$
2: **for** $j = 0, ..., J-1$
3: $\quad \Phi = \exp(-\frac{1}{\alpha}(D - 2D_p T^{(j)} D_q^\top)) \odot T^{(j)}$.
4: $\quad$ Sinkhorn iteration: $\hat{} = \frac{q}{\Phi^\top a}$, $a = \frac{p}{\Phi b}$,
5: $\quad T^{(j+1)} = \mathrm{diag}(a)\Phi\mathrm{diag}(b)$.
6: **Return** $T^{(J)}$

B. The Proof of Theorem 3.4

Theorem B.1. *For $x, y \in \mathcal{I}$, where $\mathcal{I} = \{x = [x_i], y = [y_i] \in \mathbb{R}^N \times \mathbb{R}^N | x_1 \leq ... \leq x_N, y_1 \leq ... \leq y_N\}$, the solution of the problem*

$$\min_{\sigma \in \mathcal{P}_N} \sum_{i,j} ((x_i - x_j)^2 - (y_{\sigma(i)} - y_{\sigma(j)})^2)^2 + \alpha \sum_i (x_i - y_{\sigma(i)})^2, \tag{17}$$

where $\mathcal{P}_N$ is the set of all permutation of $\{1, ..., N\}$, is invariant with respect to any translations of x and y.

Proof. Denote the translations of x and y as $x' = x + t_x 1_N$ and $y' = y + t_y 1_N$, respectively, where $t_x, t_y \in \mathbb{R}$. We then denote the objective function in (17) as $F(x, y, \sigma)$. Accordingly, we have

$$
\begin{aligned}
F(x', y', \sigma) =& \sum_{i,j} ((x_i + t_x - x_j - t_x)^2 - (y_{\sigma(i)} + t_y - y_{\sigma(j)} - t_y)^2)^2 \\
& + \alpha \sum_i (x_i + t_x - y_{\sigma(i)} - t_y)^2 \\
=& \sum_{i,j} ((x_i - x_j)^2 - (y_{\sigma(i)} - y_{\sigma(j)})^2)^2 \\
& + \alpha \sum_i \left[(x_i - y_{\sigma(i)})^2 + (t_x - t_y)^2 + 2(x_i - y_{\sigma(i)})(t_x - t_y) \right] \\
=& F(x, y, \sigma) + \alpha N(t_x - t_y)^2 + 2\alpha(t_x - t_y) \sum_i (x_i - y_{\sigma(i)}) \\
=& F(x, y, \sigma) + \underbrace{\alpha N(t_x - t_y)^2 + 2\alpha(t_x - t_y)(X - Y)}_{\text{not dependent on } \sigma},
\end{aligned} \tag{18}
$$

where $X = \sum_i x_i$, $Y = \sum_i y_{\sigma(i)} = \sum_i y_i$. Based on (18), we have

$$\min_{\sigma \in \mathcal{P}_N} F(x', y', \sigma) = \text{Constant} + \min_{\sigma \in \mathcal{P}_N} F(x, y, \sigma), \tag{19}$$

whose solution is invariant. $\square$

Theorem B.2. *Following the notations in Theorem B.1, for $x, y \in \mathcal{I}$ we denote their zero-mean translations as $x' = x + t_x 1_N$ and $y' = y + t_y 1_N$, respectively. $\arg\min_{\sigma \in \mathcal{P}_N} F(x, y, \sigma)$ satisfies:*
1) When $(\sum_i x'_i y'_i + \frac{\alpha}{8})^2 \geq (\sum_i x'_i y'_{n+1-i} + \frac{\alpha}{8})^2$, the solution is the identity permutation $\sigma(i) = i$.
2) Otherwise, the solution is the anti-identity permutation $\sigma(i) = n + 1 - i$.

Proof. The proposed problem is equivalent to the following problem

$$\max_{\sigma \in \mathcal{P}_N} Z(x, y, \sigma) = \max_{\sigma \in \mathcal{P}_N} \sum_{i,j} (x_i - x_j)^2 (y_{\sigma(i)} - y_{\sigma(j)})^2 + \alpha \sum_i x_i y_{\sigma(i)}, \tag{20}$$

and we denote $X = \sum_i x_i$ and $Y = \sum_i y_i = \sum_i y_{\sigma(i)}$. Accordingly, we have

$$
\begin{aligned}
\max_{\sigma \in \mathcal{P}_N} Z(x, y, \sigma) &= \max_{\sigma \in \mathcal{P}_N} \sum_{i,j} (x_i - x_j)^2 (y_{\sigma(i)} - y_{\sigma(j)})^2 + \alpha \sum_i x_i y_{\sigma(i)} \\
&= \max_{\sigma \in \mathcal{P}_N} \sum_{i,j} (x_i^2 + x_j^2)(y_{\sigma(i)}^2 + y_{\sigma(j)}^2) - 2 \sum_{i,j} x_i x_j (y_{\sigma(i)}^2 + y_{\sigma(j)}^2) \\
&\quad - 2 \sum_{i,j} (x_i^2 + x_j^2) y_{\sigma(i)} y_{\sigma(j)} + 4 \sum_{i,j} x_i x_j y_{\sigma(i)} y_{\sigma(j)} + \alpha \sum_i x_i y_{\sigma(i)} \\
&= \max_{\sigma \in \mathcal{P}_N} 2N \sum_i x_i^2 y_{\sigma(i)}^2 + 2 \sum_i x_i^2 \sum_i y_i^2 - 2 \sum_{i,j} x_i x_j (y_{\sigma(i)}^2 + y_{\sigma(j)}^2) \\
&\quad - 2 \sum_{i,j} (x_i^2 + x_j^2) y_{\sigma(i)} y_{\sigma(j)} + 4 \sum_{i,j} x_i x_j y_{\sigma(i)} y_{\sigma(j)} + \alpha \sum_i x_i y_{\sigma(i)} \\
&= \max_{\sigma \in \mathcal{P}_N} 2N \sum_i x_i^2 y_{\sigma(i)}^2 + 2 \sum_i x_i^2 \sum_i y_i^2 - 4X \sum_i x_i y_{\sigma(i)}^2 \\
&\quad - 4Y \sum_i x_i^2 y_{\sigma(i)} + 4 \sum_{i,j} x_i x_j y_{\sigma(i)} y_{\sigma(j)} + \alpha \sum_i x_i y_{\sigma(i)} \\
&= 2 \Big(\max_{\sigma \in \mathcal{P}_N} \sum_i N x_i^2 y_{\sigma(i)}^2 - 2X \sum_i x_i y_{\sigma(i)}^2 \\
&\quad - 2Y \sum_i x_i^2 y_{\sigma(i)} + 2(\sum_i x_i y_{\sigma(i)})^2 + \frac{\alpha}{2} \sum_i x_i y_{\sigma(i)} \Big) + 2 \sum_i x_i^2 \sum_i y_i^2,
\end{aligned}
\tag{21}
$$

where the last term $2 \sum_i x_i^2 \sum_i y_i^2$ does not depend on σ. Therefore, we define

$$W(x, y, \sigma) := \sum_i f(x_i, y_{\sigma(i)}) + \frac{\alpha}{2} \sum_i x_i y_{\sigma(i)} + 2(\sum_i x_i y_{\sigma(i)})^2, \tag{22}$$

where

$$f(x_i, y_{\sigma(i)}) := N x_i^2 y_{\sigma(i)}^2 - 2X x_i y_{\sigma(i)}^2 - 2Y x_i^2 y_{\sigma(i)}, \tag{23}$$

such that $\forall x, y \in \mathcal{I}$

$$
\begin{aligned}
\arg\max_{\sigma \in \mathcal{P}_N} Z(x, y, \sigma) &= \arg\max_{\sigma \in \mathcal{P}_N} W(x, y, \sigma) \\
&= \arg\max_{\sigma \in \mathcal{P}_N} N \sum_i f(x_i, y_{\sigma(i)}) + \frac{\alpha}{2} \sum_i x_i y_{\sigma(i)} + 2(\sum_i x_i y_{\sigma(i)})^2.
\end{aligned}
\tag{24}
$$

Furthermore, we define a translated version of $\sum_i f(x_i, y_{\sigma(i)})$ as

$$g(x, y, b) = \sum_i f(x_i + b, y_{\sigma(i)}). \tag{25}$$

According to the translation invariance shown in Theorem B.1, we assume $X = Y = 0$ and want to find a translation b^* such that $g(x, y, b^*) = 0$ and simply the problem. Specifically, we have

$$
\begin{aligned}
g(x, y, b) &= \sum_i f(x_i + b, y_{\sigma(i)}) \\
&= \sum_i N(x_i + b)^2 y_{\sigma(i)}^2 - 2(X + Nb)(x_i + b) y_{\sigma(i)}^2 - 2Y(x_i + b)^2 y_{\sigma(i)} \\
&= \sum_i N(x_i^2 + 2bx_i + b^2) y_{\sigma(i)}^2 - 2(Nbx_i + Nb^2) y_{\sigma(i)}^2 \\
&= N \Big(\sum_i x_i^2 y_{\sigma(i)}^2 - b^2 \sum_i y_{\sigma(i)}^2 \Big).
\end{aligned}
\tag{26}
$$

Obviously, $b^* = \pm \frac{\sum_i x_i^2 y_{\sigma(i)}^2}{\sum_i y_{\sigma(i)}^2}$ makes $g(x, y, b^*) = 0$. Plugging $X = Y = 0$ and b^* into (22), we have

$$
\begin{aligned}
W(x + b^* 1_N, y, \sigma) &= g(x, y, b^*) + + \frac{\alpha}{2} \sum_i (x_i + b^*) y_{\sigma(i)} + 2\Big(\sum_i (x_i + b^*) y_{\sigma(i)}\Big)^2 \\
&= \frac{\alpha}{2} \sum_i x_i y_{\sigma(i)} + 2\Big(\sum_i x_i y_{\sigma(i)}\Big)^2 \\
&= 2\Big(\sum_i x_i y_{\sigma(i)} + \frac{\alpha}{8}\Big)^2 - \frac{\alpha^2}{32}.
\end{aligned}
\tag{27}
$$

In summary, the original problem is equivalent to $\max_{\sigma \in \mathcal{P}_N} W(x + b^* 1_N, y, \sigma)$.

For any $x, y \in \mathcal{I}$, we recall the rearrangement inequality:

$$
\forall \sigma \in \mathcal{P}_N, \quad \sum_i x_i y_{n+1-i} \leq \sum_i x_i y_{\sigma(i)} \leq \sum_i x_i y_i
\tag{28}
$$

Based on (28), the solution of $\max_{\sigma \in \mathcal{P}_N} W(x + b^* 1_N, y, \sigma)$ satisfies:

- When $(\sum_i x_i y_i + \frac{\alpha}{8})^2 \geq (\sum_i x_i y_{n+1-i} + \frac{\alpha}{8})^2$, the solution is the identity permutation $\sigma(i) = i$.
- Otherwise, the solution is the anti-identity permutation $\sigma(i) = n + 1 - i$.

$\square$

In our case, $\alpha = \frac{1-\beta}{\beta}$.

C. Implementation Details

For the MNIST dataset, the architecture of the autoencoders is

$$
\begin{aligned}
\text{Encoder } Q: \quad & x \in \mathbb{R}^{28 \times 28} \to \text{Conv}_{128} \to \text{Conv}_{256} \to \text{Conv}_{512} \to \text{Conv}_{1024} \to \text{FC}_8 \to z \in \mathbb{R}^8 \\
\text{Decoder } G: \quad & z \in \mathbb{R}^8 \to \text{FC}_{7 \times 7 \times 1024} \to \text{FSConv}_{512} \to \text{FSConv}_{256} \to \text{FSConv}_1 \to x \in \mathbb{R}^{28 \times 28},
\end{aligned}
$$

where Conv_k stands for a convolution with k 4×4 filters, FSConv_k for the fractional strided convolution with k 4×4 filters, and FC_k for the fully connected layer mapping to $\mathbb{R}^k$. Except the last layer of the decoder, each of the remaining convolution layers is followed by the batch normalization and the ReLU activation. For probabilistic autoencoders, the last layer of the encoder contains two FC layers, outputting the mean and the logarithmic variance accordingly.

For the CelebA dataset, the architecture of the autoencoders is

$$
\begin{aligned}
\text{Encoder } Q: \quad & x \in \mathbb{R}^{64 \times 64 \times 3} \to \text{Conv}_{128} \to \text{Conv}_{256} \to \text{Conv}_{512} \to \text{Conv}_{1024} \to \text{FC}_{64} \to z \in \mathbb{R}^{64} \\
\text{Decoder } G: \quad & z \in \mathbb{R}^{64} \to \text{FC}_{8 \times 8 \times 1024} \to \text{FSConv}_{512} \to \text{FSConv}_{256} \to \text{FSConv}_{128} \to \text{FSConv}_3 \to x \in \mathbb{R}^{64 \times 64 \times 3},
\end{aligned}
$$

where Conv_k stands for a convolution with k 5×5 filters, FSConv_k for the fractional strided convolution with k 5×5 filters, and FC_k for the fully connected layer mapping to $\mathbb{R}^k$. Except the last layer of the decoder, each of the remaining convolution layers is followed by the batch normalization and the ReLU activation. For probabilistic autoencoders, the last layer of the encoder contains two FC layers, outputting the mean and the logarithmic variance accordingly.

For the multi-view learning experiments, the architecture of the autoencoders is

$$
\begin{aligned}
\text{Encoder } Q: \quad & x \in \mathbb{R}^V \to \text{FC}_{50} + \text{ReLU} \to \text{FC}_{20} \to z \in \mathbb{R}^{20} \\
\text{Decoder } G: \quad & z \in \mathbb{R}^{20} \to \text{FC}_{50} + \text{ReLU} \to \text{FC}_V \to x \in \mathbb{R}^V,
\end{aligned}
$$

where V is the dimension of the data. For probabilistic autoencoders, the last layer of the encoder contains two FC layers, outputting the mean and the logarithmic variance accordingly.

It should be noted that our relational co-training strategy is feasible when the latent codes of different views are with different dimensions. Here, we set the dimension as 20 for all the views to guarantee the fairness on comparisons. In such a situation, both our relational regularizer and the traditional regularizers used in AE+CoReg and AE+W are applicable. Accordingly, the comparisons among them demonstrate the superiority of our strategy.

| (a) Caltech101-7 | (b) Caltech101-20 | (c) Handwritten | (d) Cathgen |
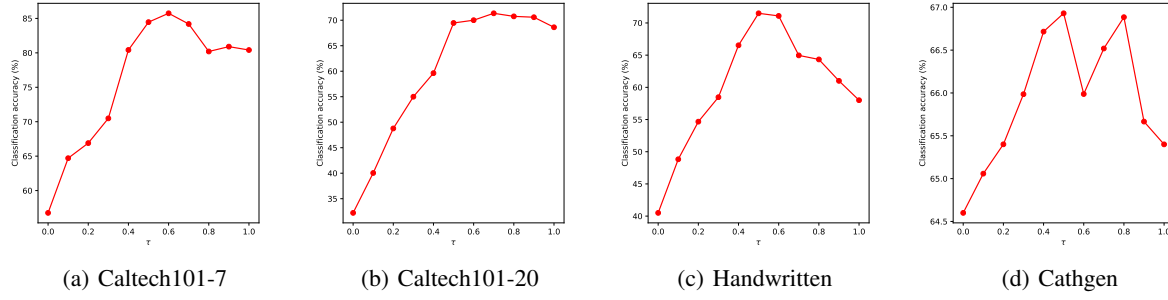
Figure 6. The influences of τ on the classification accuracy for various datasets.
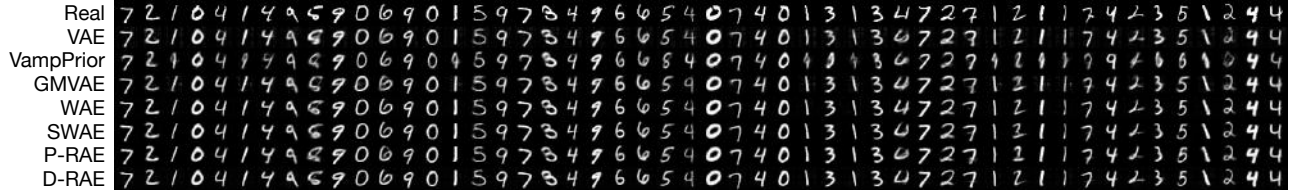
(a) MNIST

(b) CelebA

Figure 7. Comparisons for various methods on the reconstruction quality of images.

D. More Experimental Results

D.1. The influence of τ on our relational co-training

For our relational co-training strategy, the hyperparameter τ controls the strength of the relational regularization for the posterior distributions. In our experiments, when $\tau = 0$, (14) degrades to learn two autoencoders independently. When $\tau = 1$, on the contrary, (14) ignores the constraint imposed by the predefined priors (Here, we assume the priors are normal distributions). In our experiments, we empirically set $\tau = 0.5$ for all four multi-view learning datasets. In Figure **??**, we further investigate the influence of τ on the learning results. According to the investigation results, we can find that setting $\tau \in [0.4, 0.8]$ achieves the best performance in most situtations.

D.2. Visual comparisons on image generation

Figure 7 compares various methods on the quality of reconstructed testing images. We can find that the reconstruction results obtained by our probabilistic RAE (P-RAE) and deterministic RAE (D-RAE) are at least comparable to their competitors.

For the MNIST dataset, we derive the latent codes of the testing images for each autoencoder and compare different autoencoders on the t-SNE embeddings of their latent codes. Figure 8 visualize these t-SNE embeddings. For the autoencoders with structured priors (*i.e.*, VampPrior, GMVAE, P-RAE, and D-RAE), the means of the Gaussian components
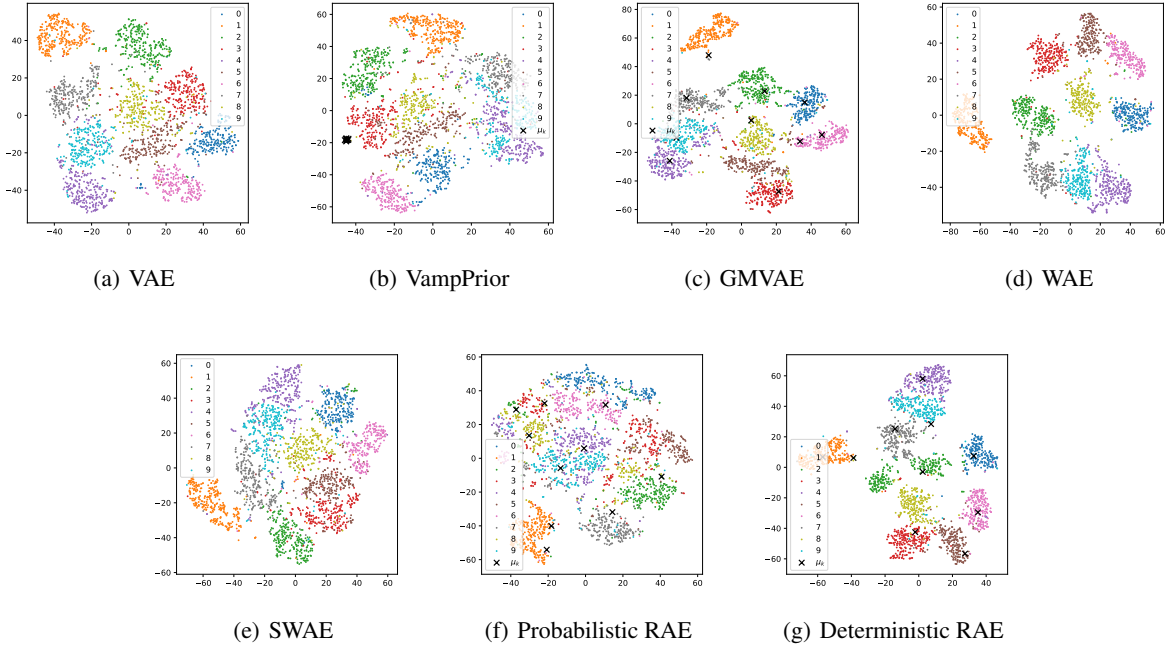
(a) VAE   (b) VampPrior   (c) GMVAE   (d) WAE

(e) SWAE   (f) Probabilistic RAE   (g) Deterministic RAE

Figure 8. Comparisons for various methods on the t-SNE embeddings of their latent codes of MNIST digits. The embeddings corresponding to different digits are with different colors. For the autoencoders using structured priors, the means of the Gaussian components are also embedded and shown as black crosses.

in their priors are embeddings as well. According to the t-SNE figures we can find that the VampPrior suffers from a serious mode collaspe problem when generating images because its Gaussian components are concentrated together, which can only represent limited modes. On the contrary, the priors of GMVAE, P-RAE, and D-RAE have diverse Gaussian components, which cover the proposed latent space well. For the CelebA dataset, we also compare various autoencoders on the t-SNE of their latent codes. In this case, we find that both GMVAE and P-RAE fail to learn structured priors — their GMM-based priors are with poor diversity. However, compared with GMVAE, which generates an undesired modality, our P-RAE at least ensures that all the Gaussian components are valid and obey to the empirical distribution of the latent codes. Our D-RAE achieves the best performance: the t-SNE results show that it successfully estimates a structured prior, whose Gaussian components are with good diversity and indicate different modalities in the latent space.

Figures 10 and 11 give more randomly generated samples. The visual effects of these samples further demonstrate the superiority of our RAEs.
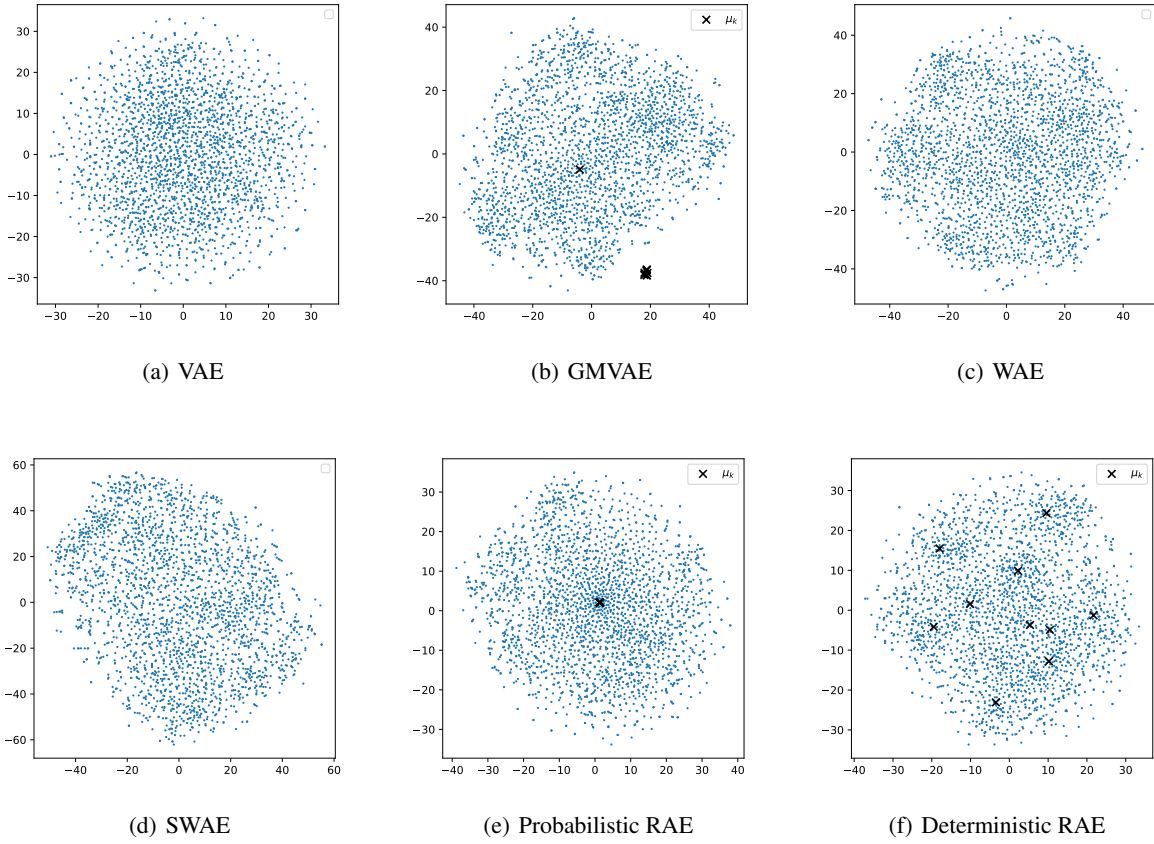
(a) VAE

(b) GMVAE

(c) WAE

(d) SWAE

(e) Probabilistic RAE

(f) Deterministic RAE

Figure 9. Comparisons for various methods on the t-SNE embeddings of their latent codes of face images. For the autoencoders using structured priors, the means of the Gaussian components are also embedded and shown as black crosses.
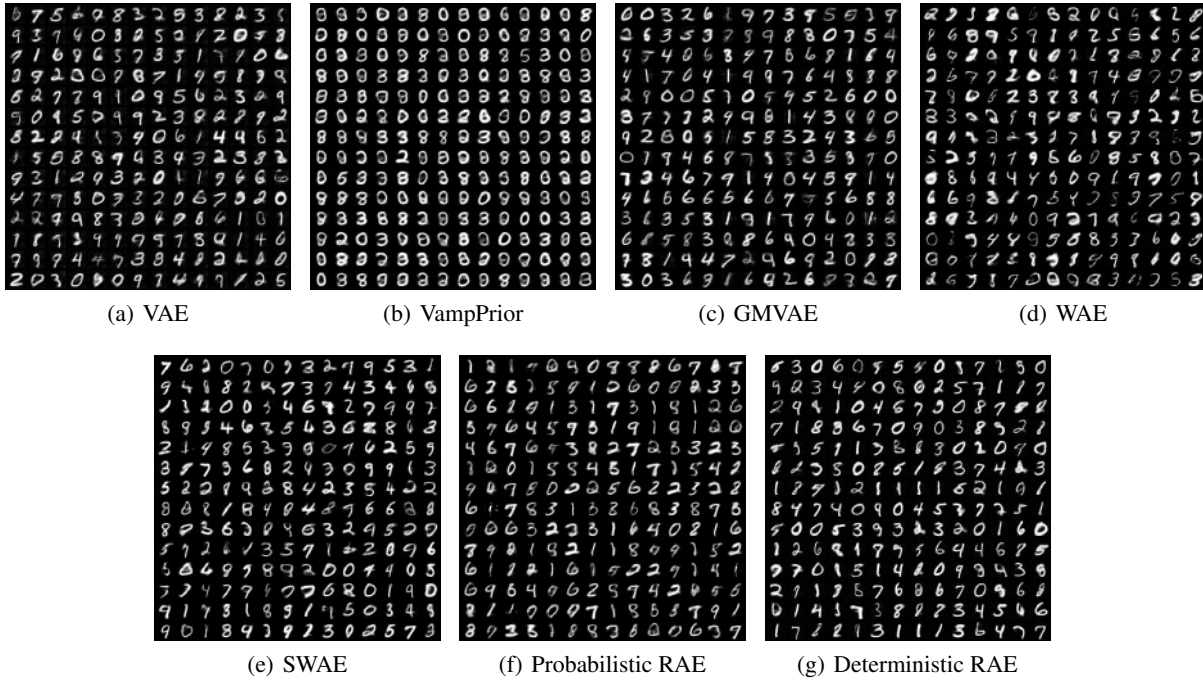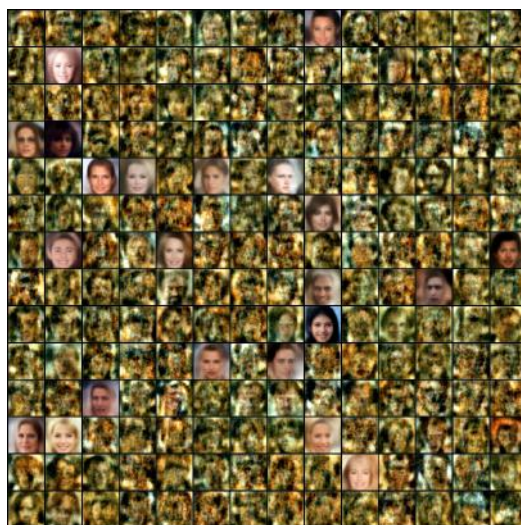

(a) VAE

(b) VampPrior

(c) GMVAE

(d) WAE

(e) SWAE

(f) Probabilistic RAE

(g) Deterministic RAE

Figure 10. Comparisons for various methods on digit generation.

(a) VAE

(b) GMVAE

(c) WAE

(d) SWAE

(e) Probabilistic RAE

(f) Deterministic RAE

Figure 11. Comparisons for various methods on face generation.