Isometric Autoencoders

Amos Gropp, Matan Atzmon & Yaron Lipman
Weizmann Institute of Science
{amos.gropp,matan.atzmon,yaron.lipman}@weizmann.ac.il

Abstract

High dimensional data is often assumed to be concentrated on or near a low-dimensional manifold. Autoencoders (AE) is a popular technique to learn representations of such data by pushing it through a neural network with a low dimension bottleneck while minimizing a reconstruction error. Using high capacity AE often leads to a large collection of minimizers, many of which represent a low dimensional manifold that fits the data well but generalizes poorly.

Two sources of bad generalization are: extrinsic, where the learned manifold possesses extraneous parts that are far from the data; and intrinsic, where the encoder and decoder introduce arbitrary distortion in the low dimensional parameterization. An approach taken to alleviate these issues is to add a regularizer that favors a particular solution; common regularizers promote sparsity, small derivatives, or robustness to noise.

In this paper, we advocate an isometry (i.e., local distance preserving) regularizer. Specifically, our regularizer encourages: (i) the decoder to be an isometry; and (ii) the encoder to be the decoder's pseudo-inverse, that is, the encoder extends the inverse of the decoder to the ambient space by orthogonal projection. In a nutshell, (i) and (ii) fix both intrinsic and extrinsic degrees of freedom and provide a non-linear generalization to principal component analysis (PCA).

Experimenting with the isometry regularizer on dimensionality reduction tasks produces useful low-dimensional data representations.

1 Introduction

A common assumption is that high dimensional data $\mathcal{X} \subset \mathbb{R}^D$ is sampled from some distribution p concentrated on, or near, some lower d-dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$, where $d < D$. The task of estimating p can therefore be decomposed into: (i) approximate the manifold $\mathcal{M}$; and (ii) approximate p restricted to, or concentrated near $\mathcal{M}$.

In this paper we focus on task (i), mostly known as *manifold learning*. A common approach to approximate the d-dimensional manifold $\mathcal{M}$, e.g., in (Tenenbaum et al., 2000; Roweis & Saul, 2000; Belkin & Niyogi, 2002; Maaten & Hinton, 2008; McQueen et al., 2016; McInnes et al., 2018), is to embed $\mathcal{X}$ in $\mathbb{R}^d$. This is often done by first constructing a graph $\mathcal{G}$ where nearby samples in $\mathcal{X}$ are conngected by edges, and second, optimizing for the locations of the samples in $\mathbb{R}^d$ striving to minimize edge length distortions in $\mathcal{G}$.

Autoencoder (AE) can also be seen as a method to learn low dimensional manifold representation of high dimensional data $\mathcal{X}$. AE is trying to reconstruct $\mathcal{X}$ as the image of its low dimensional embedding. When restricting AE to linear encoders and decoders it learns linear subspaces; with mean squared reconstruction loss they reproduce principle component analysis (PCA). Using higher capacity neural networks as the encoder and decoder, allows complex manifolds to be approximated. To avoid overfitting, different regularizers are added to the AE loss. Popular regularizers include sparsity promoting (Ranzato et al., 2007; 2008; Glorot et al., 2011), contractive or penalizing large derivatives (Rifai et al., 2011a;b), and denoising (Vincent et al., 2010; Poole et al., 2014). Recent AE regularizers directly promote distance preservation of the encoder (Pai et al., 2019; Peterfreund et al., 2020).

In this paper we advocate a novel AE regularization promoting isometry (i.e., local distance preservation), called Isometric-AE (I-AE). Our key idea is to promote the decoder to be isometric, and the encoder to be its *pseudo-inverse*. Given an isometric encoder $\mathbb{R}^d \to \mathbb{R}^D$, there is no well-defined inverse $\mathbb{R}^D \to \mathbb{R}^d$; we define the pseudo-inverse to be a projection on the image of the decoder composed with the inverse of the decoder restricted to its image.

Locally, the I-AE regularization therefore encourages: (i) the differential of the decoder $\boldsymbol{A} \in \mathbb{R}^{D\times d}$ to be an isometry, i.e., $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}_d$, where $\boldsymbol{I}_d$ is the $d \times d$ identity matrix; and (ii) the differential of the encoder, $\boldsymbol{B} \in \mathbb{R}^{d\times D}$ to be the pseudo-inverse (now in the standard linear algebra sense) of the differential of the decoder $\boldsymbol{A} \in \mathbb{R}^{D\times d}$, namely, $\boldsymbol{B} = \boldsymbol{A}^+$. In view of (i) this implies $\boldsymbol{B} = \boldsymbol{A}^T$. This means that *locally* our decoder and encoder behave like PCA, where the encoder and decoder are linear transformations satisfying (i) and (ii); That is, the PCA encoder can be seen as a composition of an orthogonal projection on the linear subspace spanned by the decoder, followed by an orthogonal transformation (isometry) to the low dimensional space.

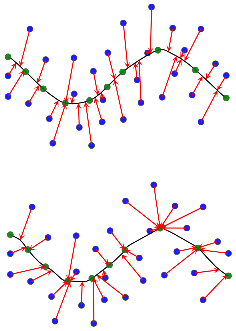

Table 1: Top: I-AE; bottom: CAE.

In a sense, our method can be seen as a version of denoising/contractive AEs (DAE/CAE, respectively). DAE and CAE promote a projection from the ambient space onto the data manifold, but can distort distances and be non-injective. Locally, using differentials again, projection on the learned manifold means $(\boldsymbol{A}\boldsymbol{B})^2 = \boldsymbol{A}\boldsymbol{B}$. Indeed, as can be readily checked conditions (i) and (ii) above imply $\boldsymbol{A}(\boldsymbol{B}\boldsymbol{A})\boldsymbol{B} = \boldsymbol{A}\boldsymbol{B}$. This means that I-AE also belongs to the same class of DAE/CAE, capturing the variations in tangent directions of the data, $\mathcal{M}$, while ignoring orthogonal variations which often represent noise (Vincent et al., 2010; Alain & Bengio, 2014). The benefit in I-AE is that its projection on the data manifold is locally an isometry, preserving distances and sampling the learned manifold evenly. The inset illustrates a simple experiment comparing contractive AE (CAE-bottom) and isometric AE (I-AE-top). Both AEs are trained on the green data points; the red arrows depict projection of points (in blue) in vicinity of the data onto the learned manifold (in black) as calculated by applying the encoder followed by the decoder. Note that CAE indeed projects on the learned manifold but not evenly, tending to shrink space around data points; in contrast I-AE provides a more even sampling of the learned manifold.

Experiments confirm that optimizing the I-AE loss results in a close-to-isometric encoder/decoder explaining the data. We further demonstrate the efficacy of I-AE for dimensionality reduction of different standard datatsets, showing its benefits over manifold learning and other AE baselines.

2 Related works

Manifold learning. Manifold learning generalize classic dimensionality reduction methods such as PCA (F.R.S., 1901) and MDS (Kruskal, 1964; Sammon, 1969), by aiming to preserve the local geometry of the data. Tenenbaum et al. (2000) use the nn-graph to approximate the geodesic distances over the manifold, followed by MDS to preserve it in the lower dimension. Roweis & Saul (2000); Belkin & Niyogi (2002); Donoho & Grimes (2003) use spectral methods to minimize different distortion energy functions over the graph matrix. Coifman et al. (2005); Coifman & Lafon (2006) approximate the heat diffusion over the manifold by a random walk over the nn-graph, to gain a robust distance measure on the manifold. Stochastic neighboring embedding algorithms (Hinton & Roweis, 2003; Maaten & Hinton, 2008) captures the local geometry of the data as a mixture of Gaussians around each data points, and try to find a low dimension mixture model by minimizing the KL-divergence. In a relatively recent work, McInnes et al. (2018) use iterative spectral and embedding optimization using fuzzy sets. Several works tried to adapt classic manifold learning ideas to neural networks and autoencoders. Pai et al. (2019) suggest to embed high dimensional points into a low dimension with a neural network by constructing a metric between pairs of data points and minimizing the metric distortion energy. Kato et al. (2019) suggest to learn an isometric decoder by using noisy latent variables. They prove under certain conditions that it encourages isometric decoder. Peterfreund et al. (2020) suggest autoencoders that promote the isometry of the encoder over the data by approximating its differential gram matrix using sample covariance matrix. Zhan et al. (2018) encourage distance preserving autoencoders by minimizing metric distortion energy in common feature space.

Modern autoencoders. There is an extensive literature on extending autoencoders to a generative model (task (ii) in section 1). That is, learning a probability distribution in addition to approximating the data manifold $\mathcal{M}$. Variational autoencoder (VAE) Kingma & Welling (2014) and its variants Makhzani et al. (2015); Burda et al. (2016); Sønderby et al. (2016); Higgins et al. (2017); Tolstikhin et al. (2018); Park et al. (2019); Zhao et al. (2019) are examples to such methods. In essence, these methods augment the AE structure with a learned probabilistic model in the low dimensional (latent) space $\mathbb{R}^d$ that is used to approximate the probability P that generated the observed data $\mathcal{X}$. More relevant to our work, are recent works suggesting regularizers for deterministic autoencoders that together with ex-post density estimation in latent space forms a generative model. Ghosh et al. (2020) suggested to reduce the decoder degrees of freedom, either by regularizing the norm of the decoder weights or the norm of the decoder differential. Other regularizers of the differential of the decoder, aiming towards a deterministic variant of VAE, were recently suggested in Kumar & Poole (2020); Kumar et al. (2020). In contrast to our method, these methods do not regularize the encoder explicitly.

3 ISOMETRIC AUTOENCODERS

We consider high dimensional data points $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^D$ sampled from some probability distribution $P(\boldsymbol{x})$ in $\mathbb{R}^D$ concentrated on or near some d dimensional submanifold $\mathcal{M} \subset \mathbb{R}^D$, where $d < D$.

Our goal is to compute *isometric autoencoder* (I-AE) defined as follows. Let $g : \mathbb{R}^D \to \mathbb{R}^d$ denote the encoder, and $f : \mathbb{R}^d \to \mathbb{R}^D$ the decoder; $\mathcal{N}$ is the learned manifold, i.e., the image of the decoder, $\mathcal{N} = f(\mathbb{R}^d)$. I-AE is defined by the following requirements:

(i) The data $\mathcal{X}$ is close to $\mathcal{N}$.

(ii) f is an isometry.

(iii) g is the pseudo-inverse of f.

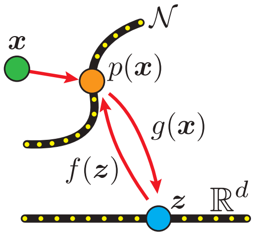

Figure 1: I-AE.

Figure 1 is an illustration of I-AE. Let θ denote the parameters of f, and ϕ the parameters of g. We enforce the requirements (i)-(iii) by prescribing a loss function $L(\theta, \phi)$ and optimize it using standard stochastic gradient descent (SGD). We next break down the loss L to its different components.

Condition (i) is promoted with the standard reconstruction loss in AE:

$$L_{\text{rec}}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^{n} \left\| f(g(\boldsymbol{x}_i)) - \boldsymbol{x}_i \right\|^2, \tag{1}$$

where $\|\cdot\|$ is the 2-norm.

Before handling conditions (ii),(iii) let us first define the notions of isometry and pseudo-inverse. A differentiable mapping f between the euclidean spaces $\mathbb{R}^d$ and $\mathbb{R}^D$ is a local isometry if it has an orthogonal differential matrix $df(\boldsymbol{z}) \in \mathbb{R}^{D \times d}$,

$$df(\boldsymbol{z})^T df(\boldsymbol{z}) = \boldsymbol{I}_d, \tag{2}$$

where $\boldsymbol{I}_d \in \mathbb{R}^{d \times d}$ is the identity matrix, and $df(\boldsymbol{z})_{ij} = \frac{\partial f^i}{\partial z_j}(\boldsymbol{z})$. A local isometry which is also a diffeomorphism is a global isometry. Restricting the decoder to isometry is beneficial for several reasons. First, Nash-Kuiper Embedding Theorem Nash (1956) asserts that non-expansive maps can be approximated arbitrary well with isometries if $D \geq d + 1$ and hence promoting an isometry does not limit the expressive power of the decoder. Second, the low dimensional representation of the data computed with an isometric encoder preserves the geometric structure of the data. In particular volume, length, angles and probability densities are preserved between the low dimensional representation $\mathbb{R}^d$, and the learned manifold $\mathcal{N}$. Lastly, for a fixed manifold $\mathcal{N}$ there is a huge space of possible decoders such that $\mathcal{N} = f(\mathbb{R}^d)$. For isometric f, this space is reduced considerably: Indeed, consider two isometries parameterizing $\mathcal{N}$, i.e., $f_1, f_2 : \mathbb{R}^d \to \mathcal{N}$. Then, since composition of isometries is an isometry we have that $f_2^{-1} \circ f_1 : \mathbb{R}^d \to \mathbb{R}^d$ is a dimension-preserving isometry and hence a rigid motion. That is, all decoders of the same manifold are the same up to a rigid motion.

For the encoder the situation is different. Since $D > d$ the encoder g cannot be an isometry in the standard sense. Therefore we ask g to be the *pseudo-inverse* of f. For that end we define the projection operator $\boldsymbol{p}$ on a submanifold $\mathcal{N} \subset \mathbb{R}^D$ as

$$\boldsymbol{p}(\boldsymbol{x}) = \underset{\boldsymbol{x}' \in \mathcal{N}}{\arg\min} \left\| \boldsymbol{x} - \boldsymbol{x}' \right\| .$$

Definition 1. *We say the g is the* pseudo-inverse *of f if g can be written as $g = f^{-1} \circ \boldsymbol{p}$, where $\boldsymbol{p}$ is the projection on $\mathcal{N} = f(\mathbb{R}^d)$.*

Consequently, if g is the pseudo-inverse of an isometry f then it extends the standard notion of isometry by projecting every point on a submanifold $\mathcal{N}$ and then applying an isometry between the d-dimensional manifolds $\mathcal{N}$ and $\mathbb{R}^d$. See Figure 1 for an illustration.

First-order characterization. To encourage f, g to satisfy the (local) isometry and the pseudo-inverse properties (resp.) we will first provide a first-order (necessary) characterization using their differentials:

Theorem 1. *Let f be a decoder and g an encoder satisfying conditions (ii),(iii). Then their differentials $\boldsymbol{A} = df(\boldsymbol{z}) \in \mathbb{R}^{D \times d}$, $\boldsymbol{B} = dg(f(\boldsymbol{z})) \in \mathbb{R}^{d \times D}$ satisfy*

$$\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I}_d \tag{3}$$

$$\boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}_d \tag{4}$$

$$\boldsymbol{B} = \boldsymbol{A}^T \tag{5}$$

The theorem asserts that the differentials of the encoder and decoder are orthogonal (rectangular) matrices, and that the encoder is the pseudo-inverse of the differential of the decoder. Before proving this theorem, let us first use it to construct the relevant losses for promoting the isometry of f and pseudo-inverse g. We need to promote conditions (3), (4), (5). Since we want to avoid computing the full differentials $\boldsymbol{A} = df(\boldsymbol{z})$, $\boldsymbol{B} = dg(f(\boldsymbol{z}))$, we will replace (3) and (4) with stochastic estimations based on the following lemma: denote the unit $d-1$-sphere by $\mathcal{S}^{d-1} = \left\{ \boldsymbol{z} \in \mathbb{R}^d | \left\| \boldsymbol{z} \right\| = 1 \right\}$.

Lemma 1. *Let $\boldsymbol{A} \in \mathbb{R}^{D \times d}$, where $d \leq D$. If $\left\| \boldsymbol{A}\boldsymbol{u} \right\| = 1$ for all $\boldsymbol{u} \in \mathcal{S}^{d-1}$, then $\boldsymbol{A}$ is column-orthogonal, that is $\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I}_d$.*

Therefore, the isometry promoting loss, encouraging (3), is defined by

$$L_{\text{iso}}(\theta) = \mathbb{E}_{\boldsymbol{z},\boldsymbol{u}} \Big(\left\| df(\boldsymbol{z})\boldsymbol{u} \right\| - 1 \Big)^2, \tag{6}$$

where $\boldsymbol{z} \sim P_{\text{iso}}(\mathbb{R}^d)$, and $P_{\text{iso}}(\mathbb{R}^d)$ is a probability measure on $\mathbb{R}^d$; $\boldsymbol{u} \sim P(\mathcal{S}^{d-1})$, and $P(\mathcal{S}^{d-1})$ is the standard rotation invariant probability measure on the $d-1$-sphere $\mathcal{S}^{d-1}$. The pseudo-inverse promoting loss, encouraging (4) would be

$$L_{\text{piso}}(\phi) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{u}} \Big(\left\| \boldsymbol{u}^T dg(\boldsymbol{x}) \right\| - 1 \Big)^2, \tag{7}$$

where $\boldsymbol{x} \sim P(\mathcal{M})$ and $\boldsymbol{u} \sim P(\mathcal{S}^{d-1})$. As-usual, the expectation with respect to $P(\mathcal{M})$ is computed empirically using the data samples $\mathcal{X}$.

Lastly, (5) might seem challenging to enforce with neural networks, however the orthogonality of $\boldsymbol{A}, \boldsymbol{B}$ can be leveraged to replace this loss with a more tractable loss asking the encoder is merely the inverse of the decoder over its image:

Lemma 2. *Let $\boldsymbol{A} \in \mathbb{R}^{D \times d}$, and $\boldsymbol{B} \in \mathbb{R}^{d \times D}$. If $\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{I}_d = \boldsymbol{B}\boldsymbol{B}^T$ and $\boldsymbol{B}\boldsymbol{A} = \boldsymbol{I}_d$ then $\boldsymbol{B} = \boldsymbol{A}^+ = \boldsymbol{A}^T$.*

Fortunately, this is already taken care of by the reconstruction loss: since low reconstruction loss in equation 1 forces the encoder and the decoder to be the inverse of one another over the data manifold, i.e.$g(f(\boldsymbol{z})) = \boldsymbol{z}$, it encourages $\boldsymbol{B}\boldsymbol{A} = \boldsymbol{I}_d$ and therefore, by Lemma 2, automatically encourages equation 5. Note that invertability also implies bijectivity of the encoder/decoder restricted to the data manifold, pushing for global isometries (rather than local). Summing all up, we define our loss for I-AE by

$$L(\theta, \phi) = L_{\text{rec}}(\theta, \phi) + \lambda_{\text{iso}} \left(L_{\text{iso}}(\theta) + L_{\text{piso}}(\phi) \right), \tag{8}$$

where λ_{iso} is a parameter controlling the isometry-reconstruction trade-off.

3.1 Details and proofs.

Let us prove Theorem 1 characterizing the relation of the differentials of isometries and pseudo-isometries, $\boldsymbol{A} = df(\boldsymbol{z}) \in \mathbb{R}^{D\times d}$, $\boldsymbol{B} = dg(f(\boldsymbol{z})) \in \mathbb{R}^{d\times D}$. First, by definition of isometry (equation 2), $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}_d$. We denote by $T_{\boldsymbol{x}}\mathcal{N}$ the d-dimensional tangent space to $\mathcal{N}$ at $\boldsymbol{x} \in \mathcal{N}$; accordingly, $T_{\boldsymbol{x}}\mathcal{N}^\perp$ denotes the normal tangent space.

Lemma 3. *The differential $d\boldsymbol{p}(\boldsymbol{x}) \in \mathbb{R}^{D\times D}$ at $\boldsymbol{x} \in \mathcal{N}$ of the projection operator $\boldsymbol{p} : \mathbb{R}^D \to \mathcal{N}$ is*

$$d\boldsymbol{p}(\boldsymbol{x})\boldsymbol{u} = \begin{cases} \boldsymbol{u} & \boldsymbol{u} \in T_{\boldsymbol{x}}\mathcal{N} \\ 0 & \boldsymbol{u} \in T_{\boldsymbol{x}}\mathcal{N}^\perp \end{cases} \tag{9}$$

That is, $d\boldsymbol{p}(\boldsymbol{x})$ is the orthogonal projection on the tangent space of $\mathcal{N}$ at $\boldsymbol{x}$.

Proof. First, consider the squared distance function to $\mathcal{N}$ defined by $\eta(\boldsymbol{x}) = \frac{1}{2}\min_{\boldsymbol{x}'\in\mathcal{N}} \|\boldsymbol{x} - \boldsymbol{x}'\|^2$. The envelope theorem implies that $\nabla\eta(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{p}(\boldsymbol{x})$. Differentiating both sides and rearranging we get $d\boldsymbol{p}(\boldsymbol{x}) = \boldsymbol{I}_D - \nabla^2\eta(\boldsymbol{x})$. As proved in Ambrosio & Soner (1994) (Theorem 3.1), $\nabla^2\eta(\boldsymbol{x})$ is the orthogonal projection on $T_{\boldsymbol{x}}\mathcal{N}^\perp$. □

Let $\boldsymbol{x} = f(\boldsymbol{z}) \in \mathcal{N}$. Since $\boldsymbol{x} \in \mathcal{N}$ we have $\boldsymbol{p}(\boldsymbol{x}) = \boldsymbol{x}$. Condition (iii) asserts that $g(\boldsymbol{y}) = f^{-1}(\boldsymbol{p}(\boldsymbol{y}))$; taking the derivative at $\boldsymbol{y} = \boldsymbol{x}$ we get $dg(\boldsymbol{x}) = df^{-1}(\boldsymbol{x})d\boldsymbol{p}(\boldsymbol{x})$. Lemma 3 implies that $d\boldsymbol{p}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{A}^T$, since $\boldsymbol{A}\boldsymbol{A}^T$ is the orthogonal projection on $\mathcal{T}_{\boldsymbol{x}}\mathcal{N}$. Furthermore, $df^{-1}(\boldsymbol{x})$ restricted to $\mathrm{Im}(\boldsymbol{A})$ is $\boldsymbol{A}^T$. Putting this together we get $\boldsymbol{B} = dg(\boldsymbol{x}) = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{A}^T$. This implies that $\boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}_d$, and that $\boldsymbol{B} = \boldsymbol{A}^+ = \boldsymbol{A}^T$. This concludes the proof of Theorem 1. □

Proof of Lemma 1. Writing the SVD of $\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$, where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_d)$ are the singular values of $\boldsymbol{A}$, we get that $\sum_{i=1}^d \sigma_i^2\boldsymbol{v}_i^2 = 1$ for all $\boldsymbol{v} \in \mathcal{S}^{d-1}$. Plugging $\boldsymbol{v} = \boldsymbol{e}_j$, $j \in [d]$ (the standard basis) we get that all $\sigma_i = 1$ for $i \in [d]$ and $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^T$ is orthogonal as claimed. □

Proof of Lemma 2. Let $\boldsymbol{U} = [\boldsymbol{A}, \boldsymbol{V}]$, $\boldsymbol{V} \in \mathbb{R}^{D\times(D-d)}$, be a completion of $\boldsymbol{A}$ to an orthogonal matrix in $\mathbb{R}^{D\times D}$. Now, $\boldsymbol{I}_d = \boldsymbol{B}\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{B}^T = \boldsymbol{I}_d + \boldsymbol{B}\boldsymbol{V}\boldsymbol{V}^T\boldsymbol{B}^T$, and since $\boldsymbol{B}\boldsymbol{V}\boldsymbol{V}^T\boldsymbol{B}^T \succeq 0$ this means that $\boldsymbol{B}\boldsymbol{V} = 0$, that is $\boldsymbol{B}$ takes to null the orthogonal space to the column space of $\boldsymbol{A}$. A direct computation shows that $\boldsymbol{B}\boldsymbol{U} = \boldsymbol{A}^T\boldsymbol{U}$ which in turn implies $\boldsymbol{B} = \boldsymbol{A}^T = \boldsymbol{A}^+$. □

Implementation. Implementing the losses in equation 6 and equation 7 requires making a choice for the probability densities and approximating the expectations. We take $P_{\text{iso}}(\mathbb{R}^d)$ to be either uniform or gaussian fit to the latent codes $g(\mathcal{X})$; and $P(\mathcal{M})$ is approximated as the uniform distribution on $\mathcal{X}$, as mentioned above. The expectations are estimated using Monte-Carlo sampling. That is, at each iteration we draw samples $\hat{\boldsymbol{x}} \in \mathcal{X}$, $\hat{\boldsymbol{z}} \sim P_{\text{iso}}(\mathbb{R}^d)$, $\hat{\boldsymbol{u}} \sim P(\mathcal{S}^{d-1})$ and use the approximations

$$L_{\text{iso}}(\theta) \approx \left(\|df(\hat{\boldsymbol{z}})\hat{\boldsymbol{u}}\| - 1 \right)^2$$
$$L_{\text{piso}}(\phi) \approx \left(\left\|\hat{\boldsymbol{u}}^T dg(\hat{\boldsymbol{x}})\right\| - 1 \right)^2$$

The right differential multiplication $df(\hat{\boldsymbol{z}})\hat{\boldsymbol{u}}$ and left differential multiplication $\hat{\boldsymbol{u}}^T dg(\hat{\boldsymbol{x}})$ are computed using forward and backward mode automatic differentiation (resp.). Their derivatives with respect to the networks' parameters θ, ϕ are computed by another backward mode automatic differentiation.

4 Experiments

4.1 Evaluation

We start by evaluating the effectiveness of our suggested I-AE regularizer, addressing the following questions: (i) does our suggested loss $L(\theta, \phi)$ in equation 8 drives I-AE training to converge to an isometry? (ii) What is the effect of the L_{piso} term? In particular, does it encourage better manifold approximations as conjectured? To that end, we examined the I-AE training on data points $\mathcal{X}$ sampled uniformly from 3D surfaces with known global parameterizations. Figure 3 shows qualitative comparison of the learned embeddings for various AE regularization techniques: Vanilla autoencoder (AE); Contractive autoencoder (CAE) (Rifai et al., 2011b); Contractive autoencoder with decoder

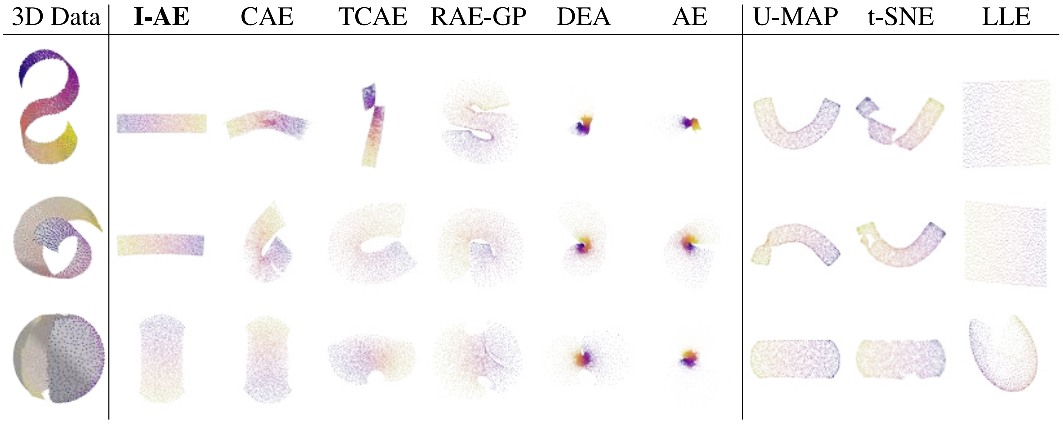

Figure 3: Evaluation of $3D \rightarrow 2D$ embeddings.

weights tied to the encoder weights (TCAE) (Rifai et al., 2011a); Gradient penalty on the decoder (RAE-GP) (Ghosh et al., 2020); and Denoising autoencoder with gaussian noise (DAE) (Vincent et al., 2010). For fairness in evaluation, all methods were trained using the same training hyper-parameters. See Appendix for the complete experiment details including mathematical formulation of the different AE regularizers. In addition, we compared versus popular classic manifold learning techniques: U-MAP (McInnes et al., 2018), t-SNE (Maaten & Hinton, 2008) and LLE. (Roweis & Saul, 2000). The results demonstrate that I-AE is able to learn an isometric embedding, showing some of the advantages in our method: sampling density and distances between input points is preserved in the learned low dimensional space.

In addition, for the AE methods, we quantitatively evaluate how close is the learnt decoder to an isometry. For this purpose, we triangulate a grid of planar points $\{\boldsymbol{z}_i\} \subset \mathbb{R}^2$. We denote by $\{e_{ij}\}$ the triangles edges incident to grid points $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$. Then, we measured the edge lengths ratio, $l_{ij} = \|f(\boldsymbol{z}_i) - f(\boldsymbol{z}_i)\|/\|e_{ij}\|$ expected to be ≈ 1 for all edges e_{ij} in an isometry. In Table 2 we log the standard deviation (Std) of $\{l_{ij}\}$ for I-AE compared to other regularized AEs. For a fair comparison, we scaled $\boldsymbol{z}_i$ so the mean of l_{ij} is 1 in all experiments. As can be seen in the table, the distribution of $\{l_{ij}\}$ for I-AE is significantly more concentrated than the different AE baselines.

	I-AE	CAE	TCAE	RAE-GP	DAE	AE
S Shape	**0.03**	0.36	0.26	1.22	2.53	1.85
Swiss Roll	**0.02**	1.00	0.38	1.75	1.80	1.63
Open Sphere	**0.07**	0.21	0.21	0.50	1.09	1.29

Table 2: Std of $\{l_{ij}\}$.

Finally, although L_{iso} is already responsible for learning an isometric decoder, the pseudo-inverse encoder (enforced by the loss L_{piso}) helps it converge to simpler solutions. We ran AE training with and without the L_{piso} term. Figure 2 shows in gray the learnt decoder surface, $\mathcal{N}$, without L_{piso} (left), containing extra (unnatural) surface parts compared to the learnt surface with L_{piso} (right). In both cases we expect (and achieve) a decoder approximating an isometry that passes through the input data points. Nevertheless, the pseudo-inverse loss restricts some of the degrees of freedom of the encoder which in turn leads to a simpler solution.

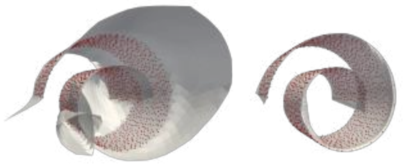

Figure 2: Decoder surfaces without L_{piso} (left) and with (right).

4.2 Data visualization

In this experiment we evaluate our method in the task of high dimension data visualization, i.e., reducing high dimensional data into two dimensions to be visually interpreted by a human observer or some downstream application. Usually the data is not assumed to lie on a manifold with such a low dimension, and it is therefore impossible to preserve all of its geometric properties. A common artifact when squeezing higher dimensional data into the plane is crowding (Maaten & Hinton, 2008), that is planar embedded points are crowded around the origin.

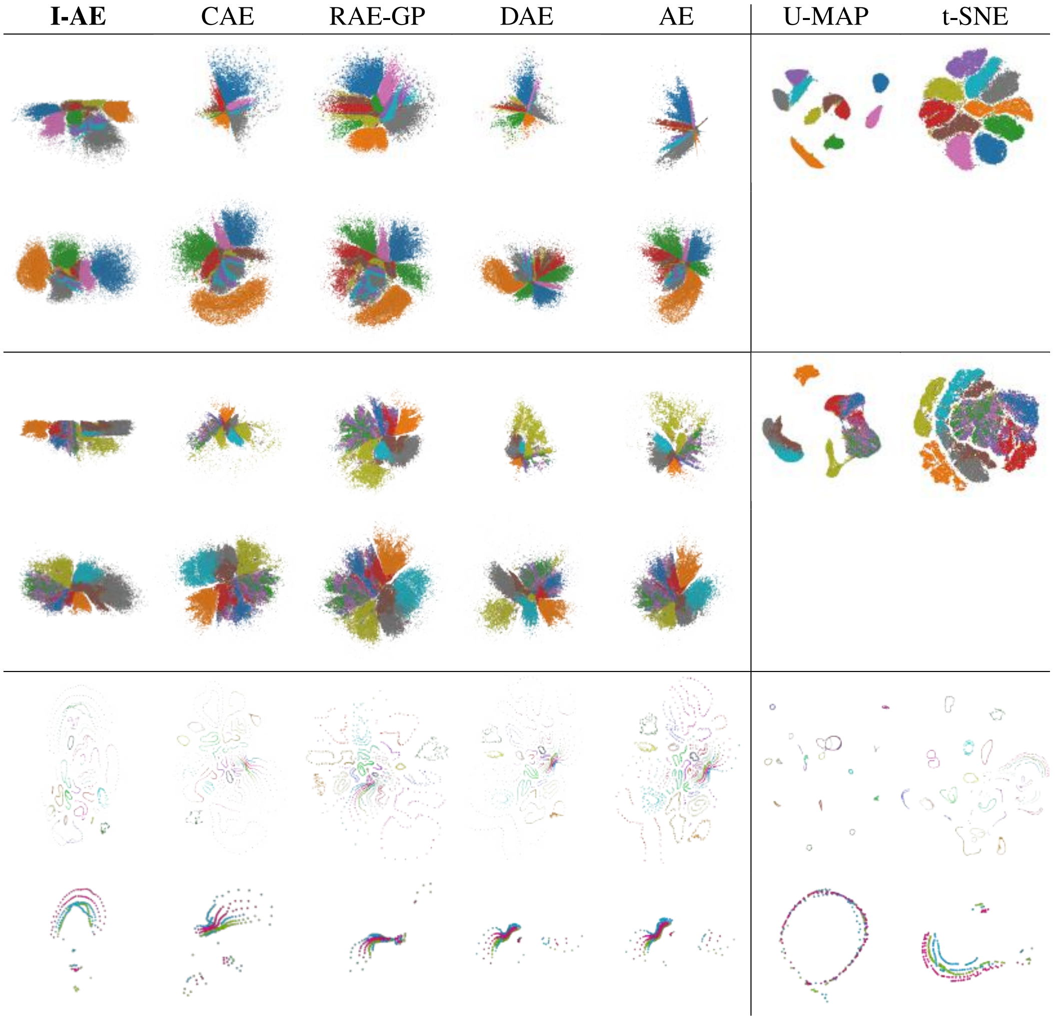

Figure 4: Results of data visualization experiment. Different colors indicate different ground turth labels/classes. Top shows MNIST: FC architecture of the encoder/decoder (top row), and CNN (bottom row); Middle shows FMNIST: FC (top row), and CNN (bottom row); Bottom shows COIL20 with CNN architecture, where zoom-ins of 3 classes are shown in the bottom row.

We evaluate our method on three standard datasets of images: MNIST (LeCun, 1998) (60k handwritten digits), Fashion-MNIST (60k Zalando's article images) (Xiao et al., 2017) and COIL20 (Nene et al., 1996) (20 different images of object rotated with 72 even rotations). For baselines we take: Vanilla AE; CAE; GP-RAE; DAE; U-MAP; and t-SNE. We use the same architecture for all auto-encoder methods on each dataset. MNIST and FMNIST we evaluated in two scenarios: (i) Both encoder and decoder are fully-connected (MLP) networks; and (ii) Both encoder and decoder are Convolutional Neural Network (CNN). For COIL20 dataset both encoder and decoder are Convolutional Neural Network. Full implementation details and hyper-parameters values can be found in the Appendix.

The results are presented in figure 4; where each embedded point z is colored by its ground-truth class/label. We make several observation. First, in all the datasets our method is more resilient to crowding compared to the baseline AEs, and provide a more even spread. U-MAP and t-SNE produce better separated clusters. However, this separation can come at a cost: See the COIL20 result (third row) and blow-ups of three of the classes (bottom row). In this dataset we expect evenly spaced points that correspond to the even rotations of the objects in the images. Note (in the blow-ups) that U-MAP maps the three classes on top of each other (non-injectivity of the "encoder"), t-SNE is somewhat better but does not preserve well the distance between pairs of data points (we expect them

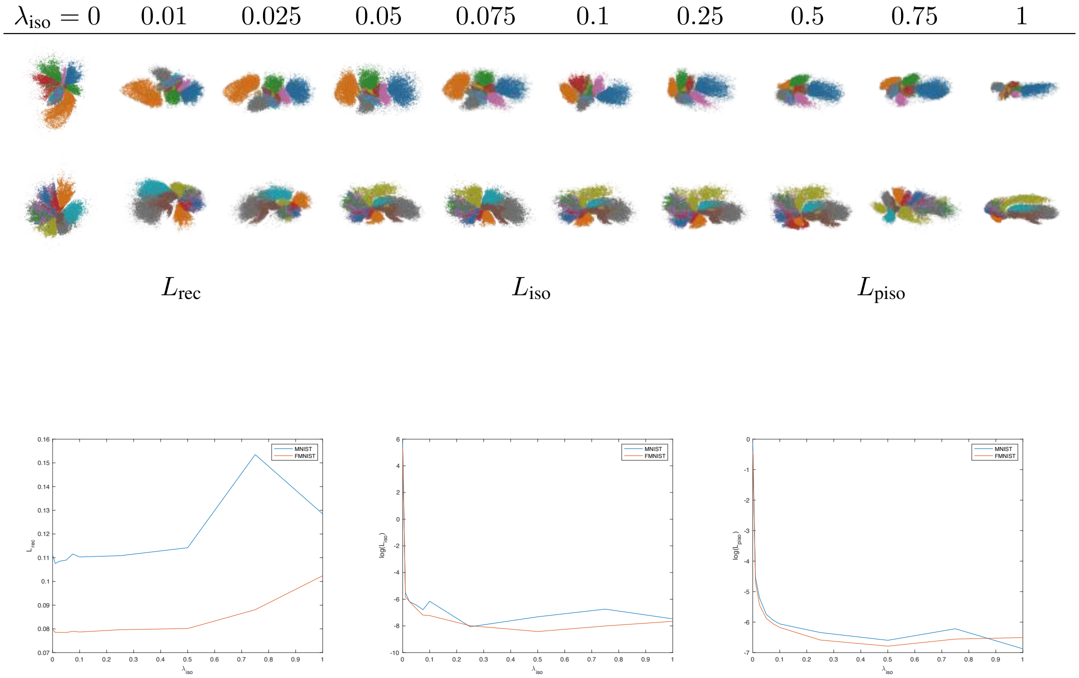

Figure 5: Sensitivity to hyper-parameters. Top: visualizations of MNIST (1st row) and FMNIST (2nd row) datasets trained with different λ_{iso} values. Bottom: plots of the final train losses as a function of λ_{iso}; left to right: L_{rec} (linear scale), L_{iso} (log scale), and L_{piso} (log scale).

to be more or less equidistant in this dataset). In I-AE the rings are better separated and points are more equidistant; the baseline AEs tend to densify the points near the origin. Lastly, considering the inter and intra-class variations for the MNIST and FMNIST datasets, we are not sure that isometric embeddings are expected to produce strongly separated clusters as in U-MAP and t-SNE (e.g., think about similar digits of different classes and dissimilar digits of the same class with distances measured in euclidean norm).

4.3 Hyper-Parameters Sensitivity

To evaluate the affect of λ_{iso} on the output we compared the visualizations and optimized loss values of MNIST and FMNIST, trained with same CNN architecture as in Section 4.2 with $\lambda_{\text{iso}} \in \{0, 0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 0.1\}$. Figure 5 shows the different visualization results as well as $L_{\text{rec}}, L_{\text{iso}}, L_{\text{piso}}$ as a function of λ_{iso}. As can be seen in both datasets the visualizations and losses are stable for λ_{iso} values between 0.01 and 0.5, where a significant change to the embedding is noticeable at 0.75. The trends in the loss values are also rather stable; L_{iso} and L_{piso} start very high in the regular AE, i.e., $\lambda_{\text{iso}} = 0$, and quickly stabilize. As for L_{rec} on FMNIST we see a stable increase while in MNIST it also starts with a steady increase until it reaches 0.75 and then it starts to be rockier, which is also noticeable in the visualizations.

5 Conclusions

We have introduced I-AE, a regularizer for autoencoders that promotes isometry of the decoder and pseudo-inverse of the encoder. Our goal was two-fold: (i) producing a favorable low dimensional manifold approximation to high dimensional data, isometrically parameterized for preserving, as much as possible, its geometric properties; and (ii) avoiding complex isometric solutions based on the notion of psuedo-inverse. Our regularizers are simple to implement and can be easily incorporated into existing autoencoders architectures. We have tested I-AE on common manifold learning tasks, demonstrating the usefulness of isometric autoencoders.

An interesting future work venue is to consider task (ii) from section 1, namely incorporating I-AE losses in a probabilistic model and examine the potential benefits of the isometry prior for generative models. One motivation is the fact that isometries push probability distributions by a simple change of coordinates, $P(\boldsymbol{z}) = P(f(\boldsymbol{z}))$.

REFERENCES

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

Luigi Ambrosio and H Mete Soner. Level set approach to mean curvature flow in arbitrary codimension. 1994.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pp. 585–591, 2002.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *CoRR*, abs/1509.00519, 2016.

Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven W Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 102(21):7426–7431, 2005.

David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003.

Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720.

Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From variational to deterministic autoencoders. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1g7tpEYDS`.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323, 2011.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pp. 6626–6637, 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

Geoffrey E Hinton and Sam T Roweis. Stochastic neighbor embedding. In *Advances in neural information processing systems*, pp. 857–864, 2003.

Keizo Kato, Jing Zhou, Tomotake Sasaki, and Akira Nakagawa. Rate-distortion optimization guided autoencoder for isometric embedding in euclidean latent space. *arXiv preprint arXiv:1910.04329*, 2019.

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

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

Abhishek Kumar and Ben Poole. On implicit regularization in β-vaes. *arXiv preprint arXiv:2002.00041*, 2020.

Abhishek Kumar, Ben Poole, and Kevin Murphy. Regularized autoencoders via relaxed injective probability flow. *arXiv preprint arXiv:2002.08927*, 2020.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

James McQueen, Marina Meila, and Dominique Joncas. Nearly isometric embedding by relaxation. In *Advances in Neural Information Processing Systems*, pp. 2631–2639, 2016.

John Nash. The imbedding problem for riemannian manifolds. *Annals of mathematics*, pp. 20–63, 1956.

Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-20). 1996.

Gautam Pai, Ronen Talmon, Alex Bronstein, and Ron Kimmel. Dimal: Deep isometric manifold learning using sparse geodesic sampling. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 819–828. IEEE, 2019.

Yookoon Park, Chris Dongjoo Kim, and Gunhee Kim. Variational laplace autoencoders. In *ICML*, 2019.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Erez Peterfreund, Ofir Lindenbaum, Felix Dietrich, Tom Bertalan, Matan Gavish, Ioannis G Kevrekidis, and Ronald R Coifman. Loca: Local conformal autoencoder for standardized data coordinates. *arXiv preprint arXiv:2004.07234*, 2020.

Ben Poole, Jascha Sohl-Dickstein, and Surya Ganguli. Analyzing noise in autoencoders and deep networks. *arXiv preprint arXiv:1406.1831*, 2014.

Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann L Cun. Efficient learning of sparse representations with an energy-based model. In *Advances in neural information processing systems*, pp. 1137–1144, 2007.

Marc'Aurelio Ranzato, Y-Lan Boureau, and Yann L Cun. Sparse feature learning for deep belief networks. In *Advances in neural information processing systems*, pp. 1185–1192, 2008.

Salah Rifai, Xavier Muller, Xavier Glorot, Grégoire Mesnil, Yoshua Bengio, and Pascal Vincent. Learning invariant features through local space contraction. *arXiv preprint arXiv:1104.4153*, 2011a.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive autoencoders: Explicit invariance during feature extraction. In *ICML*, 2011b.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

John W Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100(5):401–409, 1969.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pp. 3738–3746, 2016.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein autoencoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=HkL7n1-0b`.

Dmitry Ulyanov. Multicore-tsne. `https://github.com/DmitryUlyanov/Multicore-TSNE`, 2016.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Yibing Zhan, Jun Yu, Zhou Yu, Rong Zhang, Dacheng Tao, and Qi Tian. Comprehensive distance-preserving autoencoders for cross-modal retrieval. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, pp. 1137–1145, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356657. doi: 10.1145/3240508.3240607. URL `https://doi.org/10.1145/3240508.3240607`.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Balancing learning and inference in variational autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5885–5892, 2019.

A Appendix

A.1 Implementation details

All experiments were conducted on a Tesla V100 Nvidia GPU using PYTORCH framework Paszke et al. (2017).

A.1.1 Notations

Table 3 describes the notation for the different network layers.

Notation	Description
LIN n	Linear layer. n denotes the output dimension.
FC n	FullyConnected layer with SoftPlus ($\beta = 100$) non linear activation. n denotes the output dimension.
FC_B n	Block consisting of Lin n, followed by a batch normalization layer and SoftPlus ($\beta = 100$) non linear activation.
CONV c, k, s, p	Convolutional layer with kernel of size $k \times k$, c output channels, s stride, and p padding.
CONV_B c, k, s, p	Block consisting of CONV c, k, s, p, followed by a batch normalization layer and SoftPlus($\beta = 100$) non linear activation.
CONVT c, k, s, p	Convolutional transpose layer with kernel of size $k \times k$, c output channels, s stride, and p padding.
CONVT_B c, k, s, p	Block consisting of CONVT c, k, s, p, followed by a batch normalization layer and SoftPlus($\beta = 100$) non linear activation.

Table 3: Layers notation.

A.1.2 Evaluation

Architecture. We used an autoencoder consisted of 5 FC 256 layers followed by a Lin 2 layer for the encoder; similarly, 5 FC 256 layers followed by a Lin 3 layer were used for the decoder.

Training details. All methods were trained for a relatively long period of 100K epochs. Training was done with the Adam optimizer Kingma & Ba (2014), setting a fixed learning rate of 0.001 and a full batch. I-AE parameter was set to $\lambda_{\text{iso}} = 0.01$.

Baselines. The following regularizers were used as baselines: Contractive autoencoder (CAE) Rifai et al. (2011b); Contractive autoencoder with decoder weights tied to the encoder weights (TCAE) Rifai et al. (2011a); Gradient penalty on the decoder (RAE-GP) Ghosh et al. (2020); Denoising autoencoder with gaussian noise (DAE) Vincent et al. (2010). For both CAE, and TCAE the regularization term is $\|dg(\boldsymbol{x})\|^2$. For RAE-GP the regularization term is $\|df(\boldsymbol{z})\|^2$. For U-MAP McInnes et al. (2018), we set the number of neighbours to 30. For t-SNE Maaten & Hinton (2008), we set perplexity$= 50$.

A.1.3 Data visualization

Architecture. Table 4 lists the complete architecture details of this experiment. Both MNIST and FMNIST were trained with FC-NN and S-CNN, and COIL20 was trained with L-CNN.

FC-NN		S-CNN		L-CNN	
Encoder	Decoder	Encoder	Decoder	Encoder	Decoder
FC_B 128	FC_B 1024	Conv_B 32,4,2,1	FC 256	Conv_B 128,4,2,1	ConvT_B 2048,4,1,0
FC_B 256	FC_B 512	Conv_B 64,4,2,1	ConvT_B 128,4,1,0	Conv_B 256,4,2,1	ConvT_B 1024,4,2,1
FC_B 512	FC_B 256	Conv_B 128,4,2,1	ConvT_B 64,4,2,1	Conv_B 512,4,2,1	ConvT_B 512,4,2,1
FC_B 1024	FC_B 128	Conv_B 256,4,2,0	ConvT 32,4,2,1	Conv_B 1024,4,2,1	ConvT_B 256,4,2,1
Lin 2	Lin 784	Lin 2	ConvT 1,4,2,3	Conv_B 2048,4,2,1	ConvT_B 128,4,2,1
				Conv_B 4096,4,2,1	ConvT 1,4,2,1
				Conv 2,2,2,1	

Table 4: High dimensional visualization experiment architectures.

Training details. Training was done using Adam optimizer Kingma & Ba (2014). The rest of the training details are on table 5.

	MNIST		FMNIST		COIL20
Architecture	FC-NN	S-NN	FC-NN	S-CNN	L-CNN
Batch Size	128	512	128	512	144
λ_{iso}	0.1	0.075	0.01	0.075	0.1
Epochs	1000	500	1000	500	1000

Table 5: High dimensional visualization training details.

Baselines. The following regularizers were used as baselines: Contractive autoencoder (CAE) Rifai et al. (2011b); Gradient penalty on the decoder (RAE-GP) Ghosh et al. (2020); Denoising autoencoder with gaussian noise (DAE) Vincent et al. (2010). For CAE the regularization term is $\|dg(\boldsymbol{x})\|^2$. For RAE-GP the regularization term is $\|df(\boldsymbol{z})\|^2$. We used U-MAP McInnes et al. (2018) official implementation with random_state $= 42$, and Ulyanov (2016) multicore implementation for t-SNE Maaten & Hinton (2008) with default parameters.

A.2 Additional Experiments

A.2.1 Generalization in high dimensional space

Next, we evaluate how well our suggested isometric prior induces manifolds that generalizes well to unseen data. We experimented with three different images datasets: MNIST (LeCun, 1998); CIFAR10 (Krizhevsky et al., 2009); and CelebA (Liu et al., 2015). We quantitatively estimate methods performance by measuring the L_2 distance and the *Fréchet Inception Distance* (FID) Heusel et al. (2017) on a held out test set. For each dataset, we used the official train-test splits.

For comparison versus baselines we have selected among relevant existing AE based methods the following: Vanilla AE (AE); autoencoder trained with weight decay (AEW); Contractive autoencoder (CAE); autoencoder with spectral weights normalization (RAE-SN); and autoencoder with L_2 regularization on decoder weights (RAE-SN). RAE-L_2 and RAE-SN were recently successfully applied to this data in (Ghosh et al., 2020), demonstrating state-of-the-art performance on this task. In addition, we compare versus the Wasserstein Auto-Encoder (WAE) Tolstikhin et al. (2018), chosen as state-of-the-art among generative autoencoders.

For evaluation fairness, all methods were trained using the same training hyper-parameters: network architecture, optimizer settings, batch size, number of epochs for training and learning rate scheduling. See the appendix for specific hyper-parameters values. In addition, we generated a validation set out of the training set using 10k samples for the MNIST and CIFAR-10 experiment, whereas for the CelebA experiment we used the official validation set. For each training epoch, we evaluated the reconstruction L_2 loss on the validation set and chose the final network weights to be the one that achieves the minimum reconstruction. We experimented with two variants of I-AE regularizers: L_{piso} and $L_{\text{piso}} + L_{\text{iso}}$. Table 7 logs the results. Note that I-AE produced competitive results with the current SOTA on this task.

Architecture. For all methods, we used an autoencoder with Convolutional and Convolutional transpose layers. Table 6 lists the complete details.

MNIST		CIFAR-10		CelebA	
Encoder	Decoder	Encoder	Decoder	Encoder	Decoder
CONV_B 128, 4, 2, 1	FC 16384	CONV_B 128, 4, 2, 1	FC 16384	CONV_B 128, 5, 2, 1	FC 65536
CONV_B 256, 4, 2, 1	CONVT_B 512, 4, 2, 1	CONV_B 256, 4, 2, 1	CONVT_B 512, 4, 2, 1	CONV_B 256, 5, 2, 1	CONVT_B 512, 4, 2, 1
CONV_B 512, 4, 2, 1	CONVT_B 256, 4, 2, 1	CONV_B 512, 4, 2, 1	CONVT_B 256, 4, 2, 1	CONV_B 512, 5, 2, 1	CONVT_B 256, 4, 2, 1
CONV_B 1024, 4, 2, 1	CONVT_B 128, 4, 2, 1	CONV_B 1024, 4, 2, 1	CONVT_B 128, 4, 2, 1	CONV_B 1024, 5, 2, 1	CONVT_B 128, 4, 2, 1
LIN 16	CONVT 1, 1, 0, 0	LIN 128	CONVT 3, 1, 0, 0	LIN 128	CONVT 3, 1, 0, 0

Table 6: High dimensional generalization experiment architectures.

Training details. Training was done with the ADAM optimizer Kingma & Ba (2014), setting a learning rate of 0.0005 and batch size 100. I-AE parameter was set to $\lambda_{\text{iso}} = 0.1$.

		Methods							
Dataset	Distance	L_{piso}	$L_{\text{piso}} + L_{\text{iso}}$	AE	AEW	CAE	RAE-SN	RAE-L_2	WAE
MNIST	L_2	**0.96**	**0.99**	1.14	1.0	1.15	1.35	1.14	1.64
	FID	6.09	7.94	**4.95**	**5.59**	6.46	10.72	11.41	6.99
CIFAR-10	L_2	**20.19**	21.05	**20.16**	20.33	20.23	21.02	20.2	21.08
	FID	70.14	**56.04**	74.79	**68.71**	71.71	70.79	71.05	74.2
CelebA	L_2	20.38	**19.93**	20.51	**19.74**	20.46	20.78	20.58	20.88
	FID	**34.68**	40.73	40.53	40.00	39.52	40.45	**38.86**	38.98

Table 7: Manifold approximation quality on test images. We log the L_2 and FID distances (lower is better) from reconstructed images to the input images. The L_2 numbers are reported $*10^3$. The top performance scores are highlighted as: **First**, **Second**.

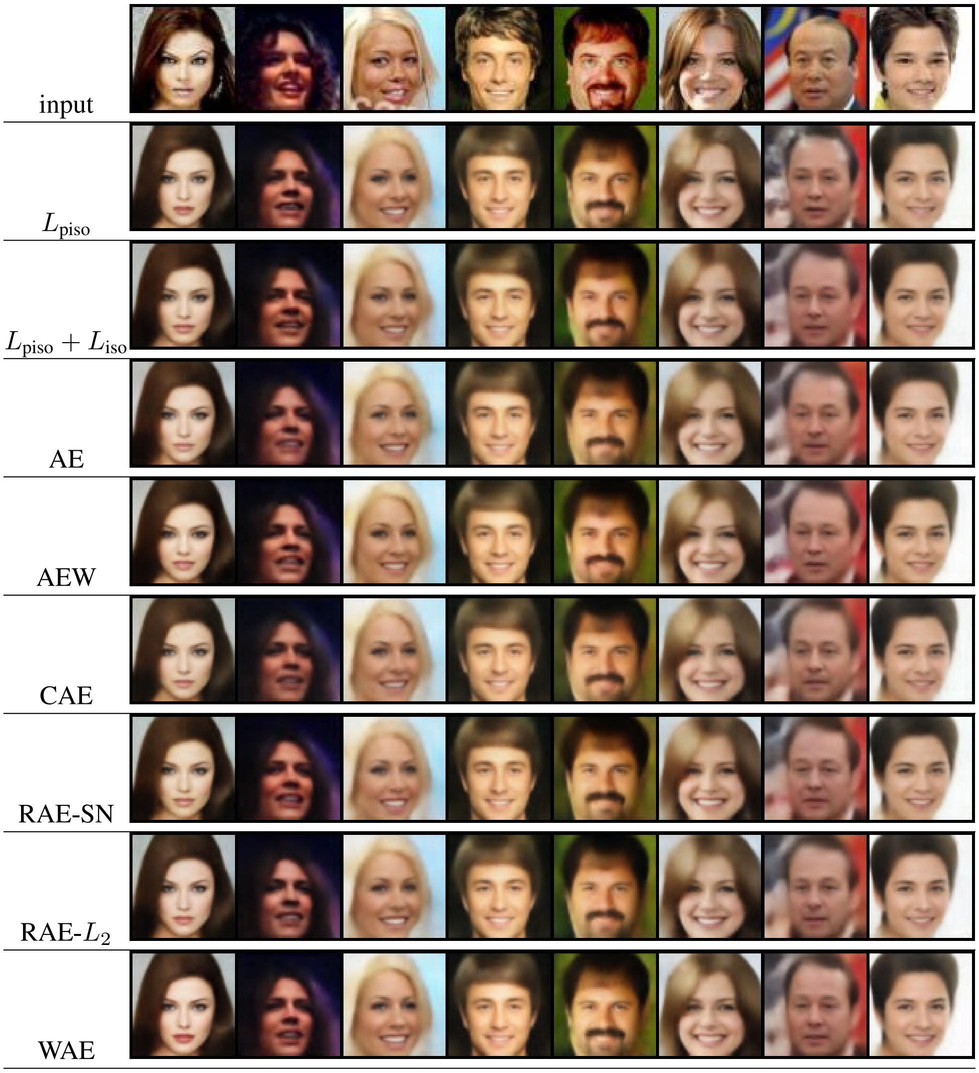

Figure 6: CelebA reconstructions.

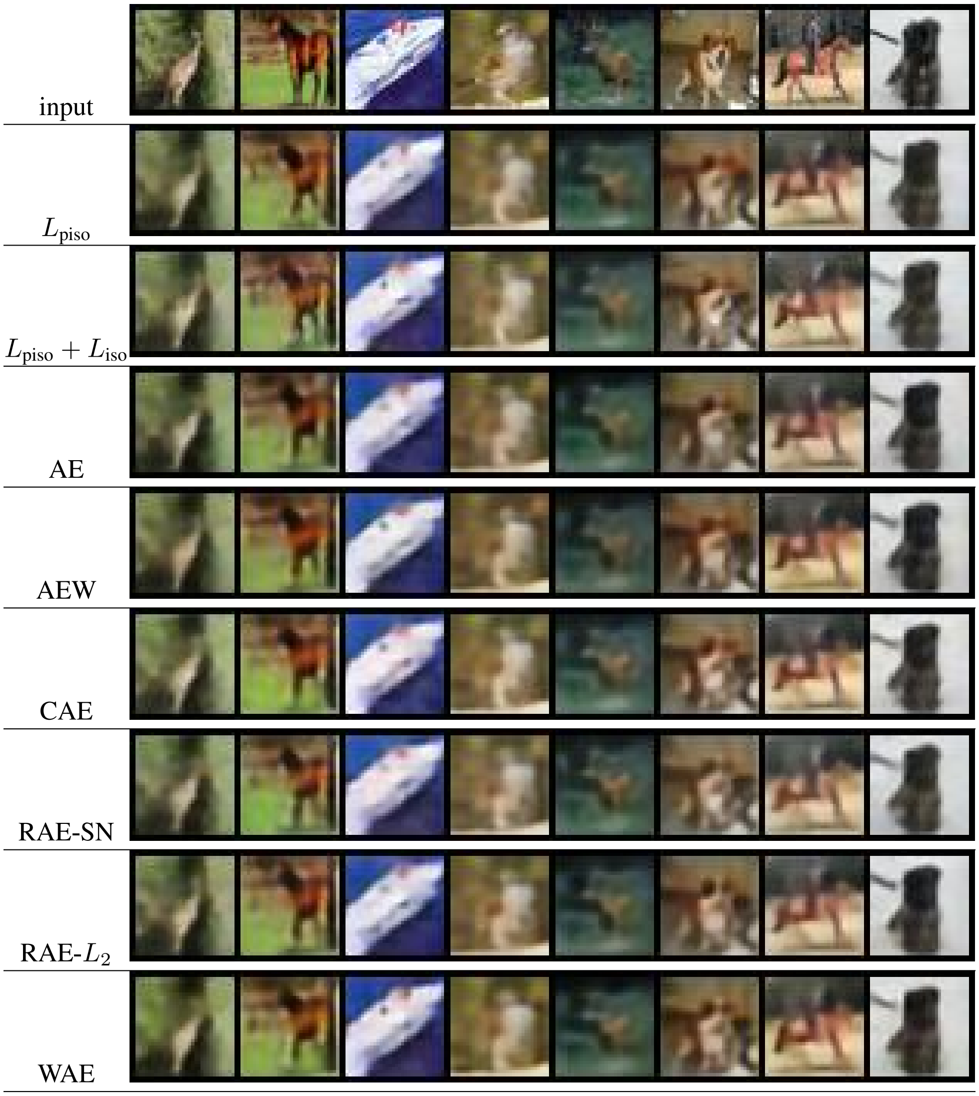

Figure 7: CIFAR-10 reconstructions.

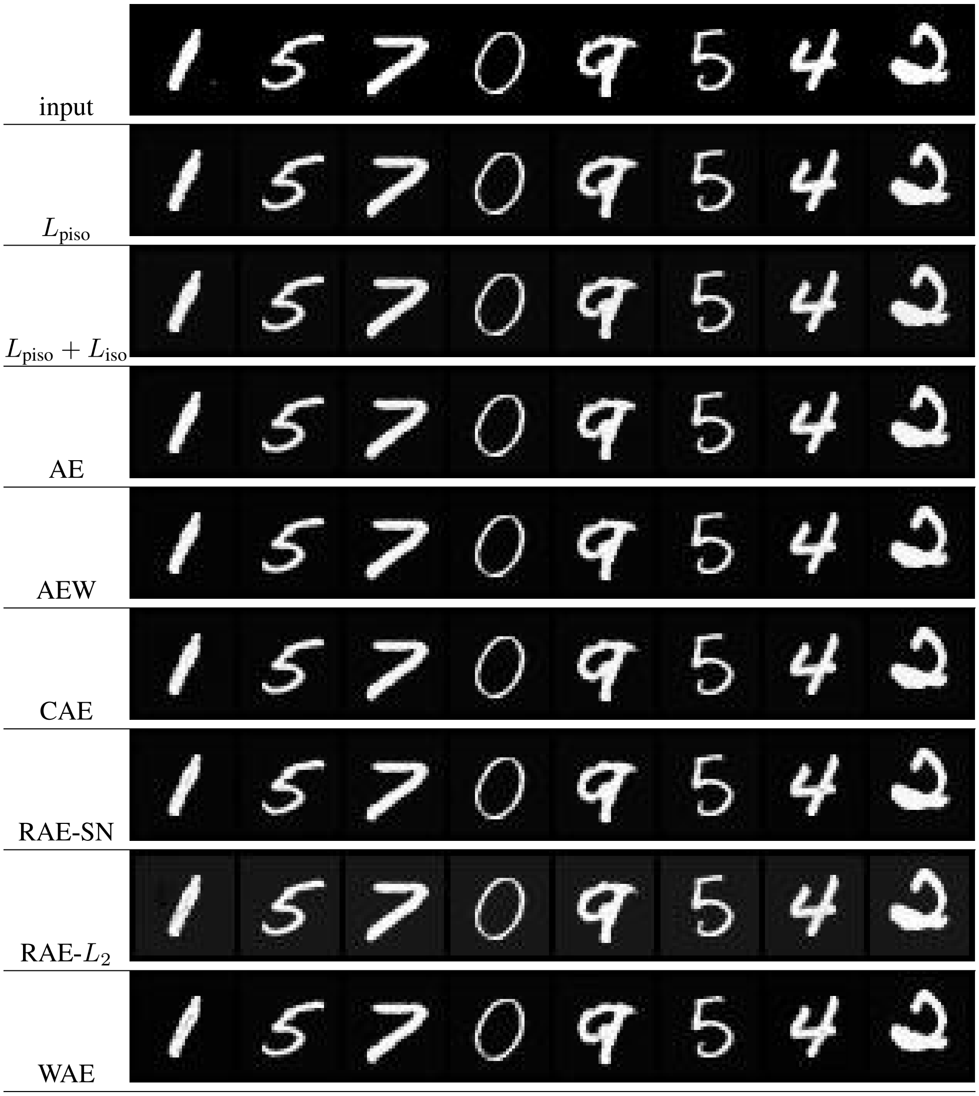

Figure 8: MNIST reconstructions.