Deep Autoencoders:
From Understanding to Generalization Guarantees

Romain Cosentino   ROM.COSENTINO@GMAIL.COM
Rice University

Randall Balestriero   RANDALLBALESTRIERO@GMAIL.COM
Rice University

Richard Baraniuk   RICHB@RICE.EDU
Rice University

Behnaam Aazhang   AAZ@RICE.EDU
Rice University

Abstract

A big mystery in deep learning continues to be the ability of methods to generalize when the number of model parameters is larger than the number of training examples. In this work, we take a step towards a better understanding of the underlying phenomena of Deep Autoencoders (AEs), a mainstream deep learning solution for learning compressed, interpretable, and structured data representations. In particular, we interpret how AEs approximate the data manifold by exploiting their continuous piecewise affine structure. Our reformulation of AEs provides new insights into their mapping, reconstruction guarantees, as well as an interpretation of commonly used regularization techniques. We leverage these findings to derive two new regularizations that enable AEs to capture the inherent symmetry in the data. Our regularizations leverage recent advances in the group of transformation learning to enable AEs to better approximate the data manifold without explicitly defining the group underlying the manifold. Under the assumption that the symmetry of the data can be explained by a Lie group, we prove that the regularizations ensure the generalization of the corresponding AEs. A range of experimental evaluations demonstrate that our methods outperform other state-of-the-art regularization techniques.

Keywords: Deep Network, Group Equivariant Network, Lie Group, Lie Algebra, Orbit, Partitioning, Generalization, Deep Autoencoders, Regression, Interpolation, Interpretability, Piecewise Linear Deep Network, Piecewise Affine Deep Network, Affine Spline Deep Network, Higher-order Regularization.

1. Introduction

Autoencoders (AEs) provide a rich and versatile framework that discovers the data's salient features in an unsupervised manner. They are commonly leveraged to efficiently perform compression (Cheng et al., 2018), denoising (Eraslan et al., 2019), data completion (Tran et al., 2017), as well as pre-training supervised DNs (Erhan et al., 2010). Solving these tasks is equivalent to discovering the data's underlying manifold, a task becoming challenging in the high dimensional and the finite samples regime (Tenenbaum et al., 2000; Wang et al., 2014, 2016; Facco et al., 2017). To overcome these challenges and improve the efficiency of AEs, various explicit or implicit regularizations have

been proposed (Vincent et al., 2008; Rifai et al., 2011b; Makhzani and Frey, 2013; Falorsi et al., 2018). Despite these improvements, the underlying mechanisms and generalization capability of AEs are still poorly understood (Li and Nguyen, 2019; Nguyen et al., 2019; Lei et al., 2020).

A compelling approach to understanding the inner mechanisms of DNs considers their capability at modeling the ubiquitous symmetries in the data (Paul and Venkatasubramanian, 2014; Mallat, 2016). Theoretically grounded models such as the Deep Scattering Network and its derivatives have been derived in accordance with this principle (Bruna and Mallat, 2013; Andén and Mallat, 2014; Chen et al., 2014; Andén et al., 2015). In Cohen and Welling (2016); Cohen et al. (2018); Kondor and Trivedi (2018) they propose to explain the success of deep convolutional architectures through the development of a theory of equivariant DNs; in particular, they provide (i) an understanding and formalism behind the equivariance properties of DNs as well as their generalization, and (ii) reduce the sample complexity of DNs by exploiting well-known symmetry group inherent to the image manifold.

Besides explicitly imposing specific group of transformations, the studies of DNs through that lens mainly consider the properties of internal layers of DNs, e.g., *convolution, pooling, per-layer representation*. In this paper, we propose a global analysis by considering the DN from a geometrical standpoint. By global analysis, we consider the understanding of the output of a DN given its input in an end-to-end manner. Such analysis is presently performed by leveraging the analytical continuous piecewise affine (CPA) map formulation of DNs, as described in Balestriero and Baraniuk (2018b). Such an approach has two significant advantages; it is agnostic of the architecture, e.g., type of layer, nonlinearities, number of layers, and it provides an analytical formula for the entire network mapping. These criteria are crucial since the understanding of AEs performed in this work has the goal of developing practical tools that are not tied to any specific AE architecture.

In the present work, the CPA formulation is leveraged to take a step into answering the following questions: (i) How an AE can effectively approximate the data manifold? (ii) How can one improve and guarantee the generalization of AEs exploiting the symmetry in the data?

We will execute this by considering the following two-fold approach: First, we provide an analytical and interpretable formulation of the CPA representation of the manifold spanned by AEs. We make explicit some critical properties of AEs such as what type of function do they belong to, how standard regularization techniques affect the AE mapping, and how the encoder and decoder per region affine mappings are related. Second, we exploit these insights to provide novel regularizations for AEs to capture the symmetry in the data. In particular, our regularizations constrain the global continuous piecewise affine surface spanned by AEs such that they adapt to the geometry of the data manifold modeled from as the orbit of a Lie group. We show that these regularizations constrain the entire surface even at locations in the manifold where data are missing, which is critical for the generalization of AEs. Besides, we show that these regularizations lead to generalization guarantees in the finite data regime.

Our contributions can be summarized as follows:

- We highlight the fact that AEs provide a CPA approximation of the data manifold. From this analytical characterization, we interpret the role of the encoder, decoder, layer parameters, and latent dimension (Sec. 3.1) as well as propose reconstruction guarantees (Sec. 3.2).

- Following these findings, we obtain interpretable formulas for the Jacobian and approximated Hessian of AEs (Sec.3.3); and leverage them to provide insights into standard regularization techniques employed in AEs (Sec. 3.4).

- We demonstrate that when considering the symmetry of the data, we can impose constraints on an interpolation function, e.g., an AE, such that it approximates the data manifold driven by a Lie group (Sec. 4.1). We turn these constraints into regularizations adapted to AEs and demonstrate their generalization guarantees under a finite data regime (Sec. 4.2).

- We finally provide experimental validations of the developed regularizations which compete with state-of-the-art methods on various datasets (Sec. 5).

2. Background and Notations

This section provides the required notions regarding the three frameworks this work intertwines.

2.1. Autoencoder

An Autoencoder (AE) aims at learning an identity mapping, also known as auto-association Ackley et al. (1985), on a given dataset with a bottleneck latent dimension. It has been implemented first for image compression Cottrell et al. (1987), speech recognition Elman and Zipser (1988), and dimensionality reduction Baldi and Hornik (1989). It is composed of two nonlinear maps: an encoder, denoted by E and a decoder, denoted by D. The encoder maps an input $x \in \mathbb{R}^d$ to a hidden layer of dimension $h < d$, $E(x)$, which encodes the salient features in the data Goodfellow et al. (2016) and defines its *code* or embedding. The decoder reconstructs the input from its *code*, thus the entire AE map is defined as $(D \circ E)(x)$ with $\circ$ denoting the composition operator.

The weights of the AE are learned based on some flavors of reconstruction losses, e.g., the mean-square error for real data and the binary cross-entropy for binary data, between the output, $(D \circ E)(x)$, and the input, x. To improve generalization, some regularizations can complement the reconstruction loss Srivastava et al. (2014) such as favoring sparsity of the *code* Makhzani and Frey (2013) or sparsity of the weights Jarrett et al.. Other types of regularization include injecting noise in the input leading to Denoising AE known to increase the robustness to small input perturbations Vincent et al. (2008). Closer to our work, Rifai et al. (2011a) and Rifai et al. (2011b) proposed to improve the robustness of the *code* to small input perturbations by penalizing the curvature of the encoder mapping by regularizing the Jacobian as well as the Hessian of E.

2.2. Max Affine Spline Network

A DN is an operator f_Θ with parameters Θ composing L intermediate *layer* mappings f_ℓ, $\ell = 1, \ldots, L$, that combine affine and simple nonlinear operators such as the *fully connected operator*, *convolution operator*, *activation operator* (applying a scalar nonlinearity such as the ubiquitous ReLU), or *pooling operator*.

A DN employing nonlinearities such as (leaky-)ReLU, absolute value, and max-pooling is a continuous piecewise linear operator and thus lives on a partition Ω of the input space. As such, the DN's CPA mapping of an input x can be written as

$$f_\Theta(x) = \sum_{\omega \in \Omega} 1_{\{x \in \omega\}} (A_\omega x + B_\omega) \tag{1}$$

where 1 defines the indicator function, A_ω and B_ω the per region affine parameters involving the DN per layer affine parameters , $W^\ell, b^\ell \in \Theta, \forall \ell$, and the nonlinearities state of the region $\omega \in \Omega$ (Balestriero and Baraniuk, 2018a). The unit and layer input space partitioning can be rewritten as Power Diagrams, a generalization of Voronoi Diagrams (Balestriero et al., 2019); composing layers produce a Power Diagram subdivision.

2.3. Lie Group Transformation

The approaches mentioned in the introduction aiming at enhancing the capability of DNs to take into account the symmetries in the data consider only those that are explicitly modeled, s.a., translation, another line of work consists in learning them from the data. In Rao and Ruderman (1999); Sohl-Dickstein et al. (2010); Hashimoto et al. (2017); Bahroun et al. (2019); Cosentino and Aazhang (2020), they propose methods capable of discovering the symmetry within the data alleviating the need for explicitly defining appropriate equivalence classes for the data. In fact, in a simple computer vision dataset such as MNIST or in a music retrieval dataset such as GTZAN, there is more than translation and rotation to characterize efficiently the data (Ick and Lostanlen, 2020). This paragraph is dedicated to the understanding of such approximation methods, which will be an important part of our regularizations.

The approximation of Lie groups has been introduced by Rao and Ruderman (1999) and later extended in Sohl-Dickstein et al. (2010); Wang et al. (2011), and aims at learning the transformation operator underlying the data with the assumption that the dataset is the result of the action of a group on a sample. This framework has an essential place in neuroscience as there is evidence of an underlying network of neurons enabling the detection of a class of equivalence via transformation learning (Kriegeskorte et al., 2008; Sengupta et al., 2018; Bahroun et al., 2019).

In the case of a Lie group, the dataset can be modeled according to the first-order Lie equation

$$\frac{d\boldsymbol{x}(\theta)}{d\theta} = G\boldsymbol{x}(\theta), \tag{2}$$

where $\boldsymbol{x}(\theta) \in \mathbb{R}^d$, θ is the coefficient governing the amount of transformation, and $G \in \mathbb{R}^{d \times d}$. This first-order differential equation indicates that the variation of the data is linear with respect to the data and depends on the infinitesimal operator $G \in \mathcal{T}_I \mathcal{G}$ where $\mathcal{T}_I \mathcal{G}$ denotes the Lie algebra of the group $\mathcal{G}$, i.e., the tangent of the group at the identity element. An introduction to group transformations can be found in Hall (2015). The solution of Eq. 2 is given by $\boldsymbol{x}(\theta) = \exp(\theta G)\boldsymbol{x}(0)$, a simple example with the group of rotation is given in Appendix C.

While the learnability of the exponential map is tedious, one can exploit its Taylor series expansion to learn the infinitesimal operator. In fact, for a small ϵ we have

$$\boldsymbol{x}(\theta + \epsilon) \approx (I + \epsilon G)\boldsymbol{x}(\theta) \tag{3}$$

The operator G can thus be learned using data that are close to each other as they result from small transformations and thus follow this approximation. Without this form of supervision, the search for neighbor data is achieved by the nearest neighbor algorithm, as in Hashimoto et al. (2017). Note that in our case, we will consider multiple transformations, each parametrized by a 1-dimensional Lie group, i.e. $\mathbf{x}(\theta) = \prod_{k=1}^{h} \exp(\theta_k G_k)\mathbf{x}(0)$, where $\theta \in \mathbb{R}^h$. In that case the first order approximation around the identity element of each group, as Eq. 3, becomes $\boldsymbol{x}(\theta + \epsilon) \approx (I + \sum_{k=1}^{h} \epsilon_k G_k)\boldsymbol{x}(\theta)$, where $\epsilon \in \mathbb{R}^h$ and with ϵ_k being the transformation parameter associated to infinitesimal operator G_k.
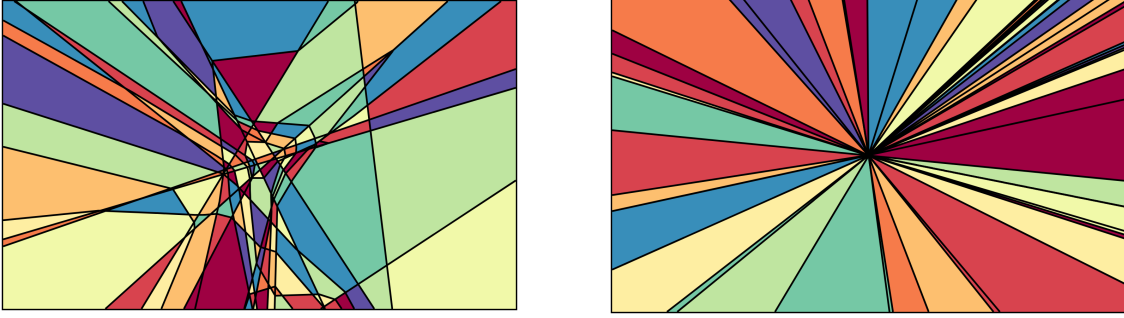
Figure 1: 2-dimensional visualizations of the input space partitioning $\Omega^{E,D}$ induced by two randomly initialized AEs with bias (*left*) and zero bias (*right*). Each region, depicted by a particular color, bounded by the black lines has a set of CPA parameters $A_\omega^E, A_\omega^D, B_\omega^E, B_\omega^D$ described in Eq. 5 which depend on the per-layer affine parameters as well as the state of the nonlinearities of the region ω. To reconstruct its input, an AE achieves an affine map for each region; its output for a sample of a given region ω is provided by Eq. 4.

3. Max Affine Spline Autoencoders

We now leverage the CPA operator defined in Eq. 1 to reformulate AEs and propose to interpret the role of the encoder and decoder (Sec. 3.1), derive a necessary condition for the reconstruction of piecewise linear data surface (Sec. 3.2), characterize its per region surface via the Jacobian and approximated Hessian of the CPA operator (Sec. 3.3), and finally use these findings to analyze commonly used variations of AEs (Sec. 3.4).

The output of a CPA DN is formed as per Eq. 1. An AE composing two CPA functions, the *encoder* and the *decoder*, the entire mapping remains a CPA with an input space partition and per region affine mappings. Because we can consider an AE as a network or as the composition of two networks, we will consider two different space partitioning. The partition of the input, i.e., data space, induced by the entire AE, and denoted by $\Omega^{E,D}$, as well as the partition of the decoder induced in the latent space, i.e., bottleneck layer, and denoted by Ω^D. Examples of the entire AE partitioning, i.e., $\Omega^{E,D}$, can be visualized in Fig. 1.

3.1. Autoencoders as a Continuous Piecewise Affine Mapping

Now, let $\omega \in \Omega^{E,D}$ defines a region induced by the AE partitioning in the input space as described in Sec. 2.2. Given a d-dimensional sample $\boldsymbol{x} \in \omega$, the max affine spline formulation of the AE mapping is defined as

$$\boldsymbol{D} \circ \boldsymbol{E}(\boldsymbol{x}) = A_\omega^D A_\omega^E \boldsymbol{x} + A_\omega^D B_\omega^E + B_\omega^D, \tag{4}$$

where $\circ$ is the composition operator, $A_\omega^D \in \mathbb{R}^{d \times h}, A_\omega^E \in \mathbb{R}^{h \times d}, B_\omega^E \in \mathbb{R}^h$ and $B_\omega^D \in \mathbb{R}^d$ with d being the dimension of the input space and h the bottleneck dimension.

The mapping from these global parameters to the per-layer ones is performed as follows. First, we denote by $W^\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}, \boldsymbol{b}^\ell \in \mathbb{R}^{d_\ell}$ the affine parameters of each layer, where $\ell \in \{1, \ldots, L\}$ defines the encoder indexes and $\ell \in \{L+1, \ldots, L+P\}$ the decoder ones (with structure depending on the layer type), where L denotes the number of encoder layers, P the number of decoder layers, $d_{\ell-1}$ the input dimension of the layer ℓ and d_ℓ its output dimension. We have that $d_L = h$ the bottleneck dimension, $d_0 = d_{L+P} = d$ the input and output dimension. Then, we also denote by

Q^ℓ the diagonal matrices encoding the region induced states of the nonlinearities, $(0, 1)$ for ReLU, $(-1, 1)$ for absolute value. Finally, the parameters of the max affine spline AE formulation described in Eq. 4 are defined as

$$A_\omega^E = W^L Q_\omega^{L-1} W^{L-1} \ldots Q_\omega^1 W^1 \quad \text{and} \quad B_\omega^E = \boldsymbol{b}^L + \sum_{i=1}^{L-1} W^L Q_\omega^{L-1} W^{L-1} \ldots Q_\omega^i \boldsymbol{b}^i. \quad (5)$$

A_ω^D and A_ω^D are defined similarly with $\ell \in \{L+1, \ldots, L+P\}$. Therefore, there is a direct mapping from the intuitive piecewise affine parameterization of the network to the per-layer parametrization as it is commonly used in the literature.

Given these analytical maps, we now provide insights into the AE approximation. Let's rewrite Eq. 4 as

$$\boldsymbol{D} \circ \boldsymbol{E}(\boldsymbol{x}) = \sum_{k=1}^h \left\langle \mathbf{a}_k^{E^T}[\omega], \boldsymbol{x} \right\rangle \mathbf{a}_k^D[\omega] + B_\omega^{E,D} = A_\omega^D \boldsymbol{\mu_x} + B_\omega^{E,D}, \quad (6)$$

where $B_\omega^{E,D} = A_\omega^D B_\omega^E + B_\omega^D$, $\mathbf{a}_k^{E^T}[\omega]$ are the rows of A_ω^E, $\mathbf{a}_k^D[\omega]$ are the columns of A_ω^D. This is the shifted mapping of $\boldsymbol{x}$ onto the subspace spanned by A_ω^D and with coordinates driven by A_ω^E.

From Eq. 6, we deduce the per region role of the encoder and decoder. The samples of each region $\omega \in \Omega^{E,D}$, are expressed in the basis defined by the decoder region-dependent parameter A_ω^D, i.e., the per region parametric representation of the approximated manifold, and the coordinates of this sample in such a basis are induced by the region-dependent parameter A_ω^E, the whole mapping is then shifted according to both the encoder and decoder CPA parameters.

3.2. Reconstruction Guarantees

We now derive a necessary condition on the CPA parameters, A_ω^D, A_ω^E, such that the AE achieves perfect reconstruction on a given continuous piecewise linear surface in the case of zero bias as often used in practice Mohan et al. (2020).

Proposition 1 *A necessary condition for the zero-bias AE to reconstruct a continuous piecewise linear data surface is to be bi-orthogonal as per $\forall \boldsymbol{x} \in \omega$, $\boldsymbol{D} \circ \boldsymbol{E}(\boldsymbol{x}) = \boldsymbol{x} \implies \left\langle \boldsymbol{a}_k^D[\omega], \boldsymbol{a}_{k'}^E[\omega] \right\rangle = \mathbb{1}_{\{k=k'\}}$. (Proof in Appendix A.1.)*

That is, if a continuous piecewise linear surface is correctly approximated, we know that the parameters of the MAS operator describing the encoder and decoder will be bi-orthogonal, i.e., the column vectors of A_ω^D and the row vectors of A_ω^E form a bi-orthogonal basis.

We now propose to give intuitions regarding this condition by utilizing the mapping between CPA parameters and layer weights as per Eq. 5. In fact, the following corollary provides the conditions for the bi-orthogonality to be fulfilled depending on the weights of the autoencoder, i.e., W^ℓ. For the sake of clarity, we consider the case of a 2-layer ReLU AE.

Corollary 1 *Let $\boldsymbol{E}$ and $\boldsymbol{D}$ be a 2-layer ReLU network with respective weights $W^1 \in \mathbb{R}^{h \times n}$ and $W^2 \in \mathbb{R}^{n \times h}$, as per Eq. 5. We denote by $W_{i,j}^1$ the ith row and j^{th} column of the weight matrix W^1. Now, $\forall x \in \mathfrak{X}$, a necessary condition for bi-orthogonality is that, for each $k, k' \in \{1, \ldots, h\}$, one of the following is fulfilled:*
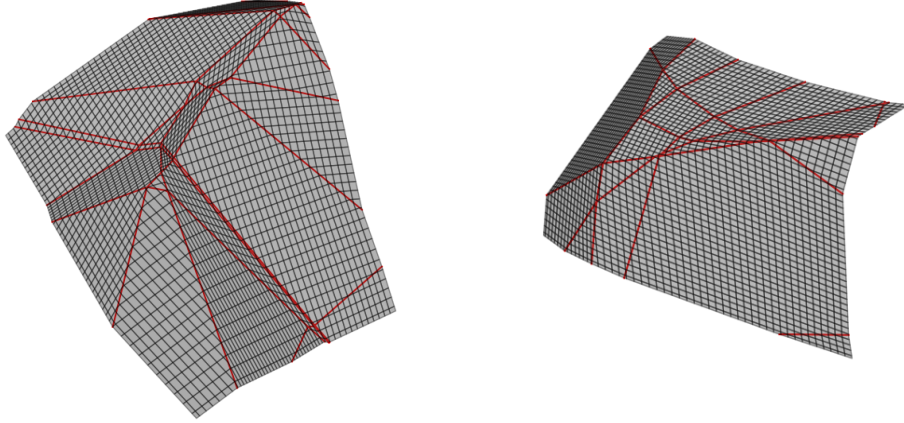
Figure 2: Piecewise linear surfaces induced by two randomly initialized AE decoders and visualized in the ambient space of dimension $d = 3$ (latent dimension being $h = 2$). The gray denotes the regions, and the red lines their borders. As they correspond to the MAS surface induced by the decoder, each gray region has a slope characterized by the Jacobien of the decoder as in Eq. 8. Our work aims at developing a constraint on these surfaces via their per region tangent, such that they approximate the manifold defined by the orbit of a signal with respect to the action of a group.

(i) $W^{1^T}_{k',.} x \leqslant 0$.

(ii) $\forall i \in \{1, \ldots, d\}, \quad W^{2^T}_{i,.} \boldsymbol{E}(x) \leqslant 0$.

(iii) $\forall i \in \{1, \ldots, d\}, \quad W^{2^T}_{i,.} \boldsymbol{E}(x) > 0$ and $\left\langle W^2_{.,k}, W^1_{k',.} \right\rangle = 0$.

(iv) $\sum_{i=1}^d W^2_{i,k} W^1_{k',i} 1_{\left\{W^{2^T}_{i,.} \boldsymbol{E}(x) > 0\right\}} = 0$.

(Proof in Appendix A.2.)

This results shows that the bi-orthogonality condition can be obtained via a combination of orthogonality conditions between the weights and/or nonlinearity activations.

For instance, the proposition (i) corresponds to the case where the input of the k' unit in the bottleneck layer is negative, condition (ii) is the case where the input of all output units is negative, condition (iii) corresponds to a linear decoder and orthogonality of the weights, and (iv) corresponds to an orthogonality condition between the kth column of the decoder weight with the k'^{th} row of the encoder weight modulo the activations of the decoder layer. Note that if (ii) and (iii) hold for multiple regions $\omega \in \Omega^{E,D}$ it implies that the decoder is linear with respect to the coordinate space and forms a linear manifold. Thus, these are not realistic conditions to have efficient AEs.

3.3. Tangents and Hessian

From the CPA formulation, we observed that for each region $\omega \in \Omega^{E,D}$, $\boldsymbol{D} \circ \boldsymbol{E}$ defines a composition of two continuous piecewise affine functions, each defined respectively by the parameters A^E_ω, B^E_ω, and A^D_ω, B^D_ω. We can thus derive simple analytical formulas for the per region Jacobian and approximated Hessian of the AE.

The Jacobian of the AE for a given region $\omega \in \Omega^{E,D}$ is given by

$$J_\omega[\boldsymbol{D} \circ \boldsymbol{E}] = A_\omega^D A_\omega^E. \tag{7}$$

More details regarding the Jacobian are given in Appendix A.7. It is also clear that the rank of the Jacobian is upper bounded by the latent dimension as $\mathrm{rank}(J_\omega[\boldsymbol{D} \circ \boldsymbol{E}]) \leqslant h$, where h is the number of units of the bottleneck layer of the AE, and in general by the $\min_\ell d_\ell$. This dimension is directly related to the manifold's dimension that one aims to approximate, assuming that all other layer widths are larger than h.

One can similarly obtain the per region tangent of the decoder, as it defines the per region parametric representation of the manifold, see Fig. 2. We recall that we denote by Ω^D the partition of the latent space induced by the decoder

$$\forall \omega \in \Omega^D, J_\omega[\boldsymbol{D}] = A_\omega^D, \tag{8}$$

where the columns of A_ω^D form the basis of the tangent space induced by $\boldsymbol{D}$.

The characterization of the curvature of the approximation of the data manifold can be done using the per region Hessian defined by $H_\omega, \forall \omega \in \Omega^D$, which in our case will be defined as the sum of the difference of neighboring tangent planes.

$$\forall \omega \in \Omega^D, \|H_\omega\|_F = \sum_{\omega' \in \mathcal{N}(\omega)} \|J_\omega[\boldsymbol{D}] - J_{\omega'}[\boldsymbol{D}]\|_F, \tag{9}$$

where $\mathcal{N}(\omega)$ denotes the set of neighbors of region ω and $\|.\|_F$ is the Frobenius norm. This approach is based on the derivation described in Rifai et al. (2011a). In practice, we use a stochastic approximation of the sum by generating a small mini-batch of a few corrupted samples which induce neighboring regions.

3.4. Interpretability of Regularization Techniques

We are now interested in leveraging these findings to analyze and interpret common AE regularizations.

(i) **Higher-Order Contractive AE** (Rifai et al., 2011a): This regularization penalizes the energy of the first and approximated second derivative the encoder map for any region containing a training sample, i.e., $\|A_\omega^E\|_F$ and $\sum_{\omega' \in \mathcal{N}(\omega)} \|A_\omega^E - A_{\omega'}^E\|_F$. In the case of a ReLU AE, we know from Eq. 4 and the submultiplicativity of the Frobenius norm that the norm of the Jacobian is upper-bounded by $\|W^L\|_F \times \cdots \times \|W^1\|_F$. Therefore adding a weight-decay penalty on the encoder weights induces the first-order contractive AE. The second-order induces the curvature of the piecewise linear map A^E to be small. Note that it is the per-region affine map induced by the encoder that is regularized, and that it depends on the region's activation codes, i.e., Q^i and $W^i \ \forall i \in \{1, \ldots, L\}$. Thus, if two neighboring regions have only have few changes in their code, and that the associated weights are small, then, such a constraint does not affect the overall curvature. On the other hand, if between two regions, the code of a unit having a weight with large amplitude does not change, then the regularization does not affect the curvature either, see details with a toy example in Appendix A.8.
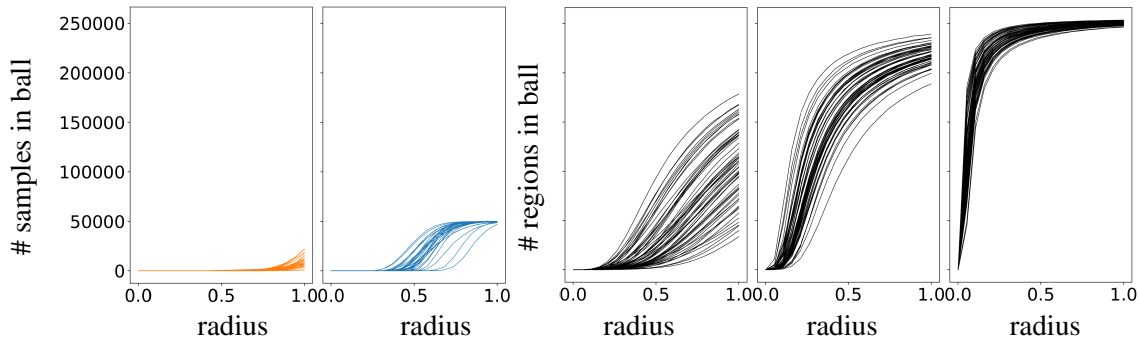
Figure 3: The first and second figures (from left to right) represent the number of data points inside a ball of growing radius (*first* to *second*: CIFAR10, MNIST). From the third to the last figure (from left to right), we show the number of regions in the latent space of the AE inside the same ball of growing radius for different AE architectures (*third* to *fifth*: Small MLP, Large MLP, Convolutional). We observe that the number of regions induced by the AE partitioning of any DN architecture in any randomly sampled ball is much larger than the number of data for any radius.

(ii) **Denoising AE** (Vincent et al., 2008): Denoising AE is known to have a similar effect than the weight-decay penalty on the DN architecture Wager et al. (2013). A penalty on the energy of W^ℓ induces a penalty on the energy of the A_ω^E and A_ω^D , $\forall \omega \in \Omega^{E,D}$. Therefore, it constrains each piece's slope to be as flat a possible, implying that the piecewise linear map focuses on approximating the low-frequency content in the data, which reinforces the learning bias of deep networks towards low-frequency information (Rahaman et al., 2018). Thus, we see how denoising and Higher-Order Contractive are tied together.

Now that we understand autoencoders' different components and their underlying functionality, we propose to constrain the surface's geometry spanned by the CPA map. In fact, we can see in Fig. 3 that for a given ball positioned in the input data space, the number of regions induced by the AE is much larger than the number of data. It is then clear that only a few of the regions contain data points. Thus, besides the implicit constraints of Deep Network, such as weight sharing on convolutional nets, and the continuity constraints of the mapping, there are no other structural constraints on the behavior of regions where no training data are available (Gamba et al., 2020; Ergen and Pilanci, 2020). *There is, therefore, a need to constrain all the regions of the CPA to guarantee the generalization capability of AEs.* The next sections aim at deriving a regularization that imposes a global structure on this mapping, which is effective even where no data points are available.

4. Generalization Via Lie Group Regularizations

For the remaining of the paper, we model the dataset as the orbit of a Lie group, that is, as per Eq. 2, $\boldsymbol{x}(\theta) = \exp(\theta G)\boldsymbol{x}(0), \theta \in \mathbb{R}, G \in \mathcal{T}_I \mathcal{G}$, where $\mathcal{T}_I \mathcal{G}$ denotes the Lie algebra of the group $\mathcal{G}$. We also assume that $\forall \theta \in \mathbb{R}, x(\theta) \in \mathbb{R}^d \backslash \{0\}$ to avoid degenerated cases. Our aim is to provide a regularization that leads to generalization guarantees, i.e., the AE is equal to **x** at any location of the manifold. In Sec. 4.1, we first provide such a regularization from a general point of view, that is, we consider the approximation of **x** by a smooth interpolation function ($C^2(\mathbb{R}, \mathbb{R}^d)$). We then

translate this condition for CPA operators (Sec. 4.2) to apply it to any AE. We then demonstrate the generalization guarantees it yields (Sec. 4.3).

4.1. Approximation of Lie Group's Orbit

First, we want to understand under which condition a interpolation function $\boldsymbol{f} \in C^2(\mathbb{R}, \mathbb{R}^d)$ coincides with the orbit of $\boldsymbol{x}(0) \in \mathbb{R}^d \setminus \{0\}$ under the action of the group $\mathcal{G}$. In particular, we propose to exploit a regularization that induces an orbit of a Lie group, such as

$$\mathcal{R}_k(\boldsymbol{f}) \triangleq \int \left\| \frac{d^k \boldsymbol{f}(\theta)}{d\theta^k} - G \frac{d^{k-1} \boldsymbol{f}(\theta)}{d\theta^{k-1}} \right\| d\theta, \tag{10}$$

where $\frac{d^k \boldsymbol{f}(\theta)}{d\theta^k}$ denotes the kth order derivative of $\boldsymbol{f}$.

This regularization constrains $\boldsymbol{f}$ such that its kth order derivative is a linear map of the $k-1$ order. In the following theorem, we show that, for $k \in \{1, 2\}$, such regularization coupled with an interpolation loss function leads to a perfect approximation of the data manifold x. That is, $\boldsymbol{f}$ coincides with $\boldsymbol{x}(\theta) = \exp(\theta G)\boldsymbol{x}(0), \forall \theta \in \mathbb{R}$ if and only if $\frac{d^k \boldsymbol{f}(\theta)}{d\theta^k} = G \frac{d^{k-1} \boldsymbol{f}(\theta)}{d\theta^{k-1}}$ and it exists a certain number of θ_i, depending on the order k, such that $\boldsymbol{f}(\theta_i) = \boldsymbol{x}(\theta_i)$. Note that the restriction to the first two orders is natural as we will apply these results on continuous piecewise affine maps, in which the second-order can only be approximated using stochastic approximation as per Sec. 3.3.

Theorem 1 *For all $k \in \{1, 2\}$, assuming G is invertible, and that a function $\boldsymbol{f}$ minimizes the regularization $\mathcal{R}_k(\boldsymbol{f})$ and it exists $\theta_i, i \in \{1, \ldots, k\}$ such that $\boldsymbol{f}(\theta_i) = \boldsymbol{x}(\theta_i)$ then $\boldsymbol{f}$ has perfect generalization as in*

$$\mathcal{R}_k(\boldsymbol{f}) = 0 \ \text{ and } \ \exists \theta_i \in \{1, \ldots, k\} \ \text{ s.t. } \ \boldsymbol{f}(\theta_i) = \boldsymbol{x}(\theta_i) \iff \forall \theta, \boldsymbol{x}(\theta) = \boldsymbol{f}(\theta). \tag{11}$$

(Proof in Appendix A.3.)

Thus an interpolant $\boldsymbol{f}$, can approximate the orbit of a Lie group, utilizing two components, the aforementioned regularization with $k \in \{1, 2\}$, and a reconstruction error that force the interpolation function to coincide with k training samples.

4.2. Lie Group Regularizations for Autoencoders

The derived regularizations were based on a smooth interpolant $\boldsymbol{f}$ and need to be adapted to the case of a CPA map. To do so, there are several crucial considerations:

(i) For the sake of clarity, the previous section illustrated the case of a one-dimensional group. Here we propose to generalize such an approach to multiple groups of transformations. We, therefore, consider the case of h infinitesimal operators $G_1, \ldots, G_h$ each corresponding to a 1-dimensional group, as explained in Sec. 2.3.

(ii) The second-order regularization requires constrains the Hessian of the CPA, which by definition, can only be approximated stochastically as explained in Sec. 3.3.

(iii) The assumption on the data is that they are generated by h transformation groups. Thus, the intrinsic dimensionality of the data is at most h. Therefore, the size of the bottleneck layer, which corresponds to the maximum dimension of the manifold the autoencoder can generate (as per Sec. 3.1), is also h.

The case $k = 1$**:** The first-order regularization corresponds to the assumption that data that are generated by the decoder and that are close to each other result from small transformations of one to another. As per Eq. 3, we obtain

$$\mathcal{R}_1(\boldsymbol{D}) \triangleq \min_{G_1,\dots,G_h} \int_{\mathbb{R}^h} \int_{\mathcal{N}(\theta)} \min_{\epsilon_1,\dots,\epsilon_h} \left\| \boldsymbol{D}(\theta) - (I + \sum_{k=1}^{h} \epsilon_k G_k) \boldsymbol{D}(\theta') \right\|_2 d\theta' d\theta, \qquad (12)$$

where $\mathcal{N}(\theta)$ denotes the neighborhood of $\theta \in \mathbb{R}^h$, the parameters $\epsilon_1, \dots, \epsilon_h$ are the scalars corresponding to the scale of the transformations, and the $G_1, \dots, G_h$ the infinitesimal operators. The optimal parameters $\epsilon^\star = [\epsilon_1, \dots, \epsilon_h]^T$ used during the training of the regularized AE are provided in Proposition 2 in Appendix B. Note that the infinitesimal operators are learned using stochastic gradient descent and that the approximation of the integrals in Eq. 12 is developed in Sec. 5.3.

The case $k = 2$**:** We know from Sec. 3.3 that for each region $\omega \in \Omega^D$, the decoder is characterized by its tangent plane, A_ω^D. The second-order regularization imposes that each tangent plane of the AE is related to their neighboring tangents plane by small transformations. Again, considering the linearized exponential maps and exploiting the definition of the Hessian in Eq. 9 we obtain the following second-order regularization on the CPA

$$\mathcal{R}_2(\boldsymbol{D}) \triangleq \min_{G_1,\dots,G_h} \int_{\mathbb{R}^h} \int_{\mathcal{N}(\omega)} \min_{\epsilon_1,\dots,\epsilon_h} \left\| J_\omega[\boldsymbol{D}] - (I + \sum_{k=1}^{h} \epsilon_k G_k) J_{\omega'}[\boldsymbol{D}] \right\|_F d\omega' d\omega, \qquad (13)$$

where $\mathcal{N}(\omega)$ denotes the set of neighbors of region ω and $\|.\|_F$ is the Frobenius norm and we recall that $J_{\omega'}[\boldsymbol{D}] = A_{\omega'}^D$ and $J_\omega[\boldsymbol{D}] = A_\omega^D$. The implementation regarding the sampling of neighboring regions is detailed Sec. 5.3. In this case also, the optimal parameters $\epsilon^\star = [\epsilon_1, \dots, \epsilon_h]^T$ used during the training of the regularized AE are provided in Proposition 3 in Appendix B.

Let us now provide interpretations regarding the Lie group regularizations we developed. While the first-order regularization constrains the AE mapping, the second-order constrains the AE's tangent plane of each region. In the first-order case the distance between $(I + \sum_{k=1}^{h} \epsilon_k G_k) \boldsymbol{D}(\theta')$, which corresponds to small transformations of the sample generated by the decoder, and $D(\theta)$ is minimized. Thus, such a regularization constrains the AE mapping to approximate the orbit induced by the infinitesimal generators. Then, the second-order regularization aims at minimizing the distance between $(I + \sum_{k=1}^{h} \epsilon_k G_k) J_{\omega'}[\boldsymbol{D}]$, which is the small transformation of the tangent plane of region ω', and $J_\omega[\boldsymbol{D}]$. This means that the second-order regularization constrains the Hessian of the decoder, which defines the angle between neighboring piecewise linear maps, to approximating the angle of the data manifold. Therefore, this penalization enforces the curvature of the piecewise linear map to fit the curvature of the orbit. Besides, as opposed to the Higher-Order Contractive AE (Rifai et al., 2011a), these regularizations constrain all the piecewise affine regions whether they contain training data or not as they do not rely on samples from the dataset. This is crucial to provide generalization guarantees in a finite data regime.

Remark: Note that, because of the first order approximation around the identity of the group, the higher order term containing the information regarding the commutativity of the group are not taken into account in our approach. The non-abelian effects are in fact of the order $O(\epsilon^2)$. Therefore, in this work, no assumptions are made on the commutativity of the group.

4.3. Approximation Error

In Sec. 4.1, we showed that if the regularization defined in Eq. 10 is equal to zero for any given $k \in \{1, 2\}$, and if the interpolation function $\boldsymbol{f}$ coincides with the data manifold defined by $\boldsymbol{x}$ on k points, then $\boldsymbol{f}$ coincides with $\boldsymbol{x}$. We now derive the generalization guarantees in the particular case where $\boldsymbol{f}$ is a CPA approximant.

Based on the assumption that (i) a region of the real manifold is correctly approximated, (ii) one of the regularizations defined in Eq. 12,13 is minimized, and that (iii) the infinitesimal operator G obtained from the regularization coincides with the infinitesimal operator of the group governing the data, we obtain the following bound on the approximation of the data manifold.

Theorem 2 *If on a region $\omega' \in \Omega^D$ the matrix $A_{\omega'}^D$ forms a basis of the manifold tangent space on this region, and it exists $k \in \{1, 2\}$ such that $\mathcal{R}_k(\boldsymbol{D}) = 0$ then for all regions $\omega \in \Omega^D$ the basis vectors of A_{ω}^D are the basis vector of the tangent of the data manifold and the distance between the continuous piecewise affine map and the data manifold is upper bounded by the radius of the regions as per*

$$d\left(\cup_{\omega \in \Omega^D} \mathcal{T}_{AE}(\omega), \mathcal{X}\right) \leqslant \sum_{\omega_i \in \Omega^D} Rad(\omega_i),$$

where $\mathcal{T}_{AE}(\omega)$ the tangent space of the AE for the region ω, $\mathcal{X}$ denotes the data manifold, d defines the 2-norm distance,and $Rad(\omega_i)$ the radius of the region ω_i. (Proof in Appendix A.6.)

The previous statement shows that if the number of pieces of the piecewise affine map, which depends on the number of neurons in the DN architecture (see Fig. 3 and refer to Montufar et al. (2014) for more details) and the type of nonlinearity, goes to infinity, then the decoder would coincide with the data manifold. In a practical setting, it tells us that the higher the number of regions is, the higher is the degrees of freedom of the CPA, and that under this regularization, these degrees of freedom are controlled while not requiring more training points.

5. Experimental Settings and Validations

In this section, we discuss some practical aspects of the proposed regularizations as well as provide the experimental validations. In particular, how the parameters of the regularizations are learned (Sec. 5.1) as well as how the sampling required in both regularizations is performed along with their induced computational complexity (Sec. 5.2), and finally the experimental validations (Sec. 5.3).

5.1. Parameters Learning

The degrees of freedom of our regularized AE comprise the usual AE parameters (per layer affine transformations) and the parameters of each regularization. The ϵ values are found from the analytical form given by Propositions 2 and 3 in Appendix B. We learn the matrices $G_k, \forall k \in \{1, \ldots, h\}$ with gradient descent based optimizer (Kingma and Ba, 2014) and thus our method introduces hd^2 additional parameters, where d is the dimension of the input data. Note that a priori knowledge on the structure of the G_k such as low-rank or skew-symmetric, i.e., Lie algebra of the special orthogonal group, can be imposed to reduce the number of parameters; we do not explore this in our study while it could be considered to speed up the computations and improve the regularization tractability. The regularizations themselves depend on the AE to find the optimal ϵ and adapt the

Table 1: Comparison of the testing reconstruction errors ($\times 10^{-2} \pm$ std $\times 10^{-2}$) for each AE (columns) and dataset (rows). The methods denoted by **Lie G. (1th)** and **Lie G. (2nd)** correspond respectively to the first-order and second-order Lie group regularizations we developed. **H.O.C. AE** denotes the Higher-Order contractive AE, and **Den. AE** denoising AE.

Dataset \ Model	AE	Den. AE	H.O.C. AE	Lie G. (1th)	Lie G. (2nd)
CIFAR10	5.6 ± 0.05	5.0 ± 0.05	-	**4.9** ± 0.07	-
MNIST	12.01 ± 0.003	12.01 ± 0.004	12.01 ± 0.004	**6.3** ± 0.1	10.13 ± 0.1
CBF	62.38 ± 0.74	52.66 ± 0.76	51.09 ± 0.54	**43.99** ± 1.2	49.73 ± 0.31
Yoga	33.76 ± 0.81	33.29 ± 0.72	32.08 ± 0.42	**20.28** ± 1.1	30.78 ± 1.2
Trace	13.95 ± 0.45	11.28 ± 0.57	12.57 ± 0.21	13.23 ± 0.4	**10.91** ± 0.45
Wine	63.06 ± 0.02	59.34 ± 0.02	49.94 ± 0.02	**19.01** ± 0.02	49.94 ± 0.01
ShapesAll	67.98 ± 3.0	58.67 ± 1.4	61.42 ± 5.5	**52.97** ± 1.9	57.80 ± 1.2
FiftyWords	64.91 ± 1.7	60.91 ± 1.0	60.92 ± 0.7	71.84 ± 3.4	**57.89** ± 1.0
WordSynonyms	70.95 ± 1.5	66.02 ± 0.8	66.52 ± 0.5	68.21 ± 2.7	**62.22** ± 1.1
InsectSounds	51.86 ± 0.6	40.24 ± 0.8	41.93 ± 0.6	**38.11** ± 0.9	38.22 ± 0.3
ECG5000	21.92 ± 0.75	20.31 ± 0.39	20.31 ± 0.36	**18.06** ± 0.9	20.29 ± 0.4
Earthquakes	56.23 ± 4.1	54.62 ± 4.1	51.79 ± 1.0	99.41 ± 0.2	**50.20** ± 0.5
Haptics	37.25 ± 0.2	36.02 ± 1.8	27.21 ± 0.5	**16.94** ± 3.4	26.06 ± 0.9
FaceFour	49.82 ± 1.0	48.51 ± 0.8	48.52 ± 0.7	48.60 ± 1.9	**46.00** ± 0.6
Synthetic	95.61 ± 1.3	89.37 ± 1.0	88.47 ± 0.9	**55.87** ± 0.8	86.83 ± 0.6

matrices G_k. The dimension of each G_k is quadratic in the dimension of the data. As such, for a high-dimensional datasets, the number of learnable parameters is large. Hence the optimization of the G_k matrices remains the current bottleneck of the method. We propose to apply the regularization term during training starting from the random initialization. More advanced strategies such as scheduled alternating minimization or employing a warm-up phase could be leveraged and result in further improvement in performance.

We now describe how we approximate the regularization terms by removing the need to evaluate the entire input space partition of the decoder for the second-order term and removing the need to sample the entire domain for the first-order term.

5.2. Region's Sampling and Computational Complexity

Recall that in the proposed second-order regularization, one should have the knowledge of the decoder latent space partition. In practice, and for large networks, the discovery of the partition would not be feasible. We thus propose to approximate the regularization by only sampling some of the regions and some of their respective neighbors. This sampling is done by first randomly sampling some vectors in the AE latent space. As for each sample, the associated per region map is automatically formed during the foward pass of the decoder, the per region parameters can be obtained by computing the affine mapping induced by the samples. To compute the neighbors of those sample regions, we use a simple dichotomic search. That is, for each of the sampled regions, we sample another (nearby) vector and keep pushing this new sample toward the first sample until one obtains the closest sample that remains in a different region. With the above, one now has the
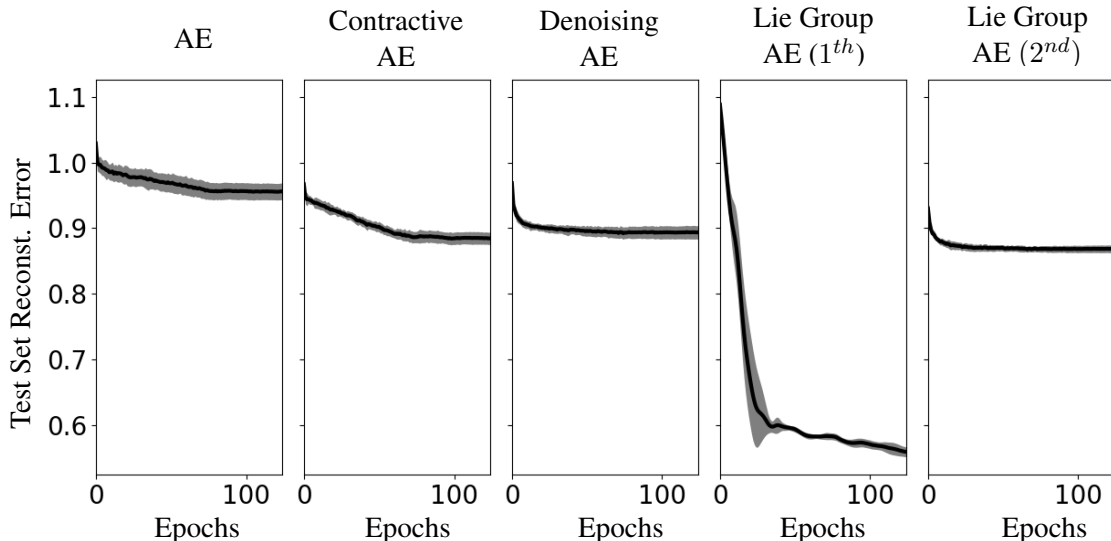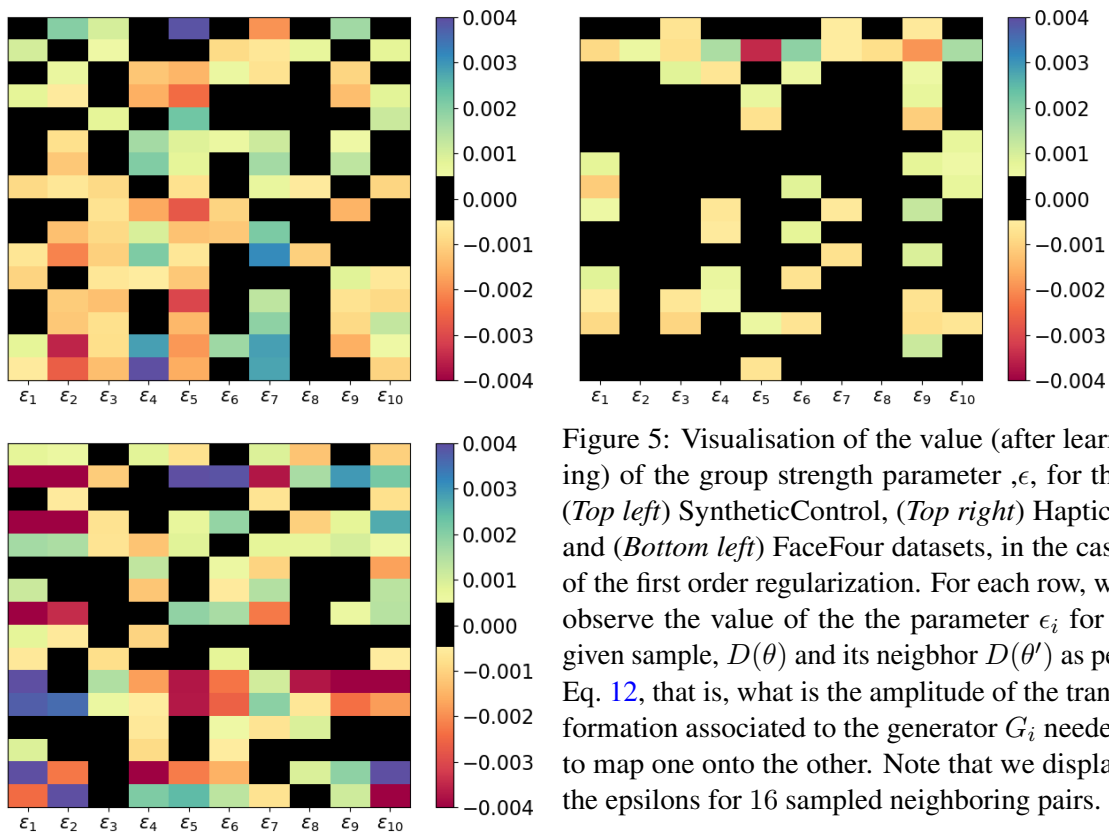
Figure 4: Test set reconstruction error on the SyntheticControl dataset evaluated on the best set of parameters for different AEs (from left to right): AE, Higher Order Contractive AE, Denoising AE, Lie Group AE (first-order), and Lie Group AE (second-order). For each model, the mean over 10 runs is reported in black, and the gray area corresponds to its standard deviation. We observe that the first order regularization performs much better than the second order one, which is close to the higher-order contactive AE error. In fact, this dataset contains six classes of time-series trends (upward, downward, normal,...), which can be easily related by a linear transformations, that is, the first order regularization can be easily optimized. The second order is harder to train and is more sensitive to the sampling of the regions, therefore is less reliable and harder to interpret.

knowledge of some regions and one neighboring region for each of those regions. We leverage this approach and perform the search of a single neighbor; for a better approximation of the regularization, one can repeat this sampling process and accumulate the obtained regions and neighbors. For the first-order term, we propose a similar approximation where we approximate the integral by sampling a latent space vector θ (at each mini-batch).

Let us now consider the computational complexity induced by the regularizations omitting the computational cost of a pass through the AE as it is shared across all techniques. The optimal coordinates ϵ^* are obtained by solving a linear system of h equations in both cases for each sampled datum (first-order) or each sampled region (second-order). This equation has to be solved for each sample region or latent space vector; we denote this by N as in our case, we sample in each mini-batch as many vectors/regions as the size of the mini-batch. We obtain the time complexity $\mathcal{O}(h^2 N + d^2 hN)$ for the first-order, and $\mathcal{O}(h^2 N + d^2 h^2 N)$ for the second-order, and a space complexity of $\mathcal{O}(d^2 h)$ in both cases being driven by the need to retain the matrices $G_1, \ldots, G_h$. The current bottleneck is the storage of those matrices, which limits the size of the AE bottleneck and output dimension.

5.3. Experiments

We evaluate our framework on diverse datasets, including images and time-series data including speech, medical as well a seismic recordings, the description of the datasets is given in Appendix D.

Figure 5: Visualisation of the value (after learning) of the group strength parameter ,ϵ, for the (*Top left*) SyntheticControl, (*Top right*) Haptics, and (*Bottom left*) FaceFour datasets, in the case of the first order regularization. For each row, we observe the value of the the parameter ϵ_i for a given sample, $D(\theta)$ and its neigbhor $D(\theta')$ as per Eq. 12, that is, what is the amplitude of the transformation associated to the generator G_i needed to map one onto the other. Note that we display the epsilons for 16 sampled neighboring pairs.

For each model and each hyperparameter, we perform 10 runs for 125 epochs with batch size 16. The results are reported in Table 1. In this table, the statistics reported correspond to the average over the 10 runs, each run using the test set performances based on the best validation set measure. Note that for CIFAR10, the computational burden of both the second-order Lie group regularization and the higher-order contractive one is too high. Thus only the AE, denoising AE and the first-order Lie group regularization are evaluated.

We propose, in particular, to visualize the test set reconstruction for the different AE models during training in Fig. 4 (see Fig. 6 in Appendix E for supplementary figures), where we can see that both Lie Group AEs are robust to the DN initialization and do not overfit. Besides, we can observe that while the first order on this dataset outperforms all the other regularizations, its variance at the begining of the learning phase is more volatile than other approaches.

The hyperparameter responsible for the variance of the noise added to the data in the Denoising AE case also corresponding to the noise added to the data to sample Jacobian of nearby regions in Higher-Order Contractive AE parameter is evaluated for the values $\{0.001, 0.01, 0.1, 1\}$. Another hyperparameter is the regularization trade-off parameter for both the Higher-Order Contractive AE and Lie Group AEs, the following values are tested for both models $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. All the models were trained using the same AE with 3 fully connected encoder layers with ReLU with bottleneck dimension $h = 10$, and 3 fully connected decoder layer with ReLU and 1 linear fully connected output layer.

We can observe in Table 1 that the Lie group regularizations are usually outperforming the other methods the different datasets we evaluated. While the second-order regularization is more computationally demanding, it appears to be more stable and robust to the change of parameters. Besides, we can see that the first-order regularization might be more sensitive to how close to a Lie group the dataset under evaluation is. In fact, both the first and second-order depends on the learned infinitesimal operator, however, while the second-order use such matrix to constrain the overall curvature of the CPA the first-order use it to constrains its mapping.

In Fig. 5, we show, for three datasets, the value of the parameters $\epsilon_i, \forall i \in \{1, \ldots, h\}$ obtained by minimizing Eq. 12. This observation is important as the framework we propose assumes that the dimension of the orbit, hence the number of group transformations, is the same as the dimension of the bottleneck layer of the autoencoder. In particular, we propose to highlight the sparsity of the parameters, i.e., if for various sampled pairs, the value a particular strength parameter, ϵ_i, is close to 0. If it is, then the associated group of transformation is not being used to map any $\mathbf{D}(\theta)$ to a close sample $\mathbf{D}(\theta')$. Therefore, the number of group of transformations selected is too large, e.g., in the top right subplot, corresponding to the Haptics dataset, we observe that, the 10 transformations are not required.

6. Conclusion

We analyzed AEs from a geometrical standpoint and provided insights into how AEs are approximating the data manifold. In particular, we provided analytical formulas of the per region map that AEs are performing using its continuous piecewise affine formulation. This approach's strength lies in its interpretability power, as for a given region in the input space, the DN mapping is a simple affine map. Leveraging these key features, we proposed to enhance and guarantee the generalization capability of AEs by proposing two regularizations that capture the symmetry in the data. These regularizations constrain the piecewise continuous surface spanned by the decoder to approximate the orbit of a Lie group. Besides, inspired by the theory of learning Lie group transformations, we alleviated the need to explicitly define a group of symmetry underlying the data and propose to learn the group's generator. In fact, the generator of a Lie group lives in a vector space, thus enabling common matrix manipulations required to perform its update.

Finally, we determined a computationally efficient procedure to learn these regularizations by performing a stochastic sampling approach to determine the per region parameters. Note that, in practice, this sampling is the bottleneck of our method since the regions do not form a regular tiling of the space. We showed that even using such an approximation, the method outperforms other existing regularizations and compete with state-of-the-art results on various datasets.

We believe our methodology opens the door to novel ways for DN practitioners to regularize the DN's parameter space depending on data assumptions. In particular, this approach should be relevant in the case of the application of DNs in science, where the underlying physical phenomenons can be partially theoretically prescribed, and following our approach, problem-specific regularization can be derived to provide the AE with generalization guarantees and improved approximation capabilities.

References

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

J. Andén and S. Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014.

J. Andén, V. Lostanlen, and S. Mallat. Joint time-frequency scattering for audio classification. *CoRR*, abs/1512.02125, 2015.

A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31:606–660, 2017.

Y. Bahroun, D. Chklovskii, and A. Sengupta. A similarity-preserving network trained on transformed images recapitulates salient features of the fly motion detection circuit. In *Advances in Neural Information Processing Systems*, pages 14178–14189, 2019.

P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

R. Balestriero and R. Baraniuk. A spline theory of deep learning. In *International Conference on Machine Learning*, pages 374–383, 2018a.

R. Balestriero and R. G. Baraniuk. Mad max: Affine spline insights into deep learning. *arXiv preprint arXiv:1805.06576*, 2018b.

R. Balestriero, R. Cosentino, B. Aazhang, and R. Baraniuk. The geometry of deep networks: power diagram subdivision. In *Advances in Neural Information Processing Systems*, pages 15806–15815, 2019.

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

X. Chen, X. Cheng, and S. Mallat. Unsupervised deep haar scattering on graphs. In *Advances in Neural Information Processing Systems*, pages 1709–1717, 2014.

Z. Cheng, H. Sun, M. Takeuchi, and J. Katto. Deep convolutional autoencoder-based lossy image compression. In *2018 Picture Coding Symposium*, pages 253–257. IEEE, 2018.

T. Cohen and M. Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

T. Cohen, M. Geiger, J. Köhler, and M. Welling. Spherical CNNs. *CoRR*, abs/1801.10130, 2018.

R. Cosentino and B. Aazhang. Learnable group transform for time-series. In *International Conference on Machine Learning*, pages 2164–2173, 2020.

G. Cottrell, P. Munro, and D. Zipser. Image compression by back propagation: An example of extensional progamming. *ICS Report*, (8702), 1987.

L. Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

J. L. Elman and D. Zipser. Learning the hidden structure of speech. *The Journal of the Acoustical Society of America*, 83(4):1615–1626, 1988.

G. Eraslan, L. M. Simon, M. Mircea, N. S. Mueller, and F. J. Theis. Single-cell rna-seq denoising using a deep count autoencoder. *Nature communications*, 10(1):1–14, 2019.

T. Ergen and M. Pilanci. Convex geometry of two-layer relu networks: Implicit autoencoding and interpretable models. In *International Conference on Artificial Intelligence and Statistics*, pages 4024–4033, 2020.

D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

E. Facco, M. d'Errico, A. Rodriguez, and A. Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):1–8, 2017.

L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. *arXiv preprint arXiv:1807.04689*, 2018.

M. Gamba, S. Carlsson, H. Azizpour, and M. Björkman. Hyperplane arrangements of trained convnets are biased. 2020.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

B. Hall. *Lie Groups, Lie Algebras, and Representations: an Elementary Introduction*, volume 222. Springer, 2015.

T. B. Hashimoto, P. S. Liang, and J. C. Duchi. Unsupervised transformation learning via convex relaxations. In *Advances in Neural Information Processing Systems*, pages 6875–6883, 2017.

C. Ick and V. Lostanlen. Learning a lie algebra from unlabeled data pairs. 2020.

K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE.

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

R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. 2018.

N. Kriegeskorte, M. Mur, D. A. Ruff, R. Kiani, J. Bodurka, H. Esteky, K. Tanaka, and P. A. Bandettini. Matching categorical object representations in inferior temporal cortex of man and monkey. *Neuron*, 60(6): 1126–1141, 2008.

A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

N. Lei, D. An, Y. Guo, K. Su, S. Liu, Z. Luo, S. Yau, and X. Gu. A geometric understanding of deep learning. *Engineering*, 2020.

P. Li and P. M. Nguyen. On random deep weight-tied autoencoders: Exact asymptotic analysis, phase transitions, and implications to training. In *International Conference on Learning Representations*, 2019.

A. Makhzani and B. Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

S. Mallat. Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150203, 2016.

S. Mohan, Z. Kadkhodaie, E. P. Simoncelli, and C. Fernandez-Granda. Robust and interpretable blind image denoising via bias-free convolutional neural networks. *arXiv preprint arXiv:1906.05478*, 2020.

G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, 27:2924–2932, 2014.

T. V. Nguyen, Raymond K. W. Wong, and C. Hegde. On the dynamics of gradient descent for autoencoders. In *Proceedings of Machine Learning Research*, volume 89, pages 2858–2867. PMLR, Apr 2019.

A. Paul and S. Venkatasubramanian. Why does deep learning work? a perspective from group theory. *arXiv preprint arXiv:1412.6621*, 2014.

N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. A Hamprecht, Y. Bengio, and A. Courville. On the spectral bias of neural networks. *arXiv preprint arXiv:1806.08734*, 2018.

R. Rao and D. L. Ruderman. Learning lie groups for invariant visual perception. In *Advances in neural information processing systems*, pages 810–816, 1999.

S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, and X. Glorot. Higher order contractive auto-encoder. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 645–660. Springer, 2011a.

S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *International Conference on Machine Learning*, 2011b.

A. Sengupta, C. Pehlevan, M. Tepper, A. Genkin, and D. Chklovskii. Manifold-tiling localized receptive fields are optimal in similarity-preserving neural networks. In *Advances in Neural Information Processing Systems*, pages 7080–7090, 2018.

J. Sohl-Dickstein, C. M. Wang, and B. A. Olshausen. An unsupervised algorithm for learning lie group transformations. *arXiv preprint arXiv:1001.1027*, 2010.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

L. Tran, X. Liu, J. Zhou, and R. Jin. Missing modalities imputation via cascaded residual autoencoder. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1405–1414, 2017.

P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

S. Wager, S. Wang, and P. S. Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013.

C. M. Wang, J. Shol-Dickstein, I. Tosic, and B. A. Olshausen. Lie group transformation models for predictive video coding. In *2011 Data Compression Conference*, pages 83–92. IEEE, 2011.

W. Wang, Y. Huang, Y. Wang, and L. Wang. Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 490–497, 2014.

Y. Wang, H. Yao, and S. Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242, 2016.

Appendix A. Proofs

A.1. Proof of Proposition 1

Proof Perfect reconstruction $\Rightarrow$: $\forall \omega, \forall x \in \omega$, $x = \sum_{k=1}^{h} \langle x, a_k^E[\omega] \rangle a_k^D[\omega]$. We have $\forall \omega, \forall x \in \omega$

$$
\sum_k \langle x, a_k^E[\omega] \rangle a_k^D[\omega] = \sum_{k=1}^{h} \left\langle \sum_{k'=1}^{h} \langle x, a_{k'}^E[\omega] \rangle a_{k'}^D[\omega], a_k^E[\omega] \right\rangle a_k^D[\omega]
$$

$$
= \sum_k \sum_{k'=1}^{h} \langle x, a_{k'}^E[\omega] \rangle \langle a_{k'}^D[\omega], a_k^E[\omega] \rangle a_k^D[\omega]
$$

$\iff A_\omega^D A_\omega^E x = A_\omega^D A_\omega^{D^T} A_\omega^{E^T} A_\omega^E x$ since $A_\omega^D A_\omega^E$ is injective on the region (as per perfect reconstruction condition) it implies that $A_\omega^{D^T} A_\omega^{E^T} = I_h$, where I_h is the identity matrix of dimension $h \times h$ ∎

A.2. Proof of Corrolary 1

Proof For a 2-layers ReLU autoencoder network, we have the following affine spline parameters $\forall x \in \omega$:

$$
a_{k'}^E[\omega] = 1_{\left\{ W_{k',.}^{1^T} x > 0 \right\}} W_{k',.}^1
$$

$$
a_k^D[\omega] = \begin{pmatrix} 1_{\left\{ W_{1,.}^{2^T} z > 0 \right\}} \\ \vdots \\ 1_{\left\{ W_{d,.}^{2^T} z > 0 \right\}} \end{pmatrix} \cdot W_{.,k}^2
$$

where $\cdot$ defines here the elementwise vector multiplication. Now,

$$
\langle a_k^D[\omega], a_{k'}^E[\omega] \rangle = \left\langle Q_\omega^2 W_{.,k}^2, 1_{\left\{ W_{k',.}^{1^T} x > 0 \right\}} W_{k',.}^1 \right\rangle
$$

$$
= 1_{\left\{ W_{k',.}^{1^T} x > 0 \right\}} W_{.,k}^{2^T} Q_\omega^2 W_{k',.}^1
$$

$$
= 1_{\left\{ W_{k',.}^{1^T} x > 0 \right\}} W_{.,k}^{2^T} \begin{pmatrix} 1_{\left\{ W_{1,.}^{2^T} E(x) > 0 \right\}} W_{k',1}^1 \\ \vdots \\ 1_{\left\{ W_{n,.}^{2^T} E(x) > 0 \right\}} W_{k',d}^1 \end{pmatrix}
$$

$$
= 1_{\left\{ W_{k',.}^{1^T} x > 0 \right\}} \left(\sum_{i=1}^{d} W_{i,k}^2 W_{k',i}^1 1_{\left\{ W_{i,.}^{2^T} E(x) > 0 \right\}} \right)
$$

∎

A.3. Proof of Theorem 1

Proof For both cases, we recall that we assume that $\forall \theta, x(\theta) \neq 0$. In fact, relaxing such assumption would lead to a degenerated case where the interpolant can be constant and equal to 0. In practice this assumption is more than realistic as the '0-datum' is usually not part of any dataset. Let's first consider the case $k = 1$.

We know that the solution of $\frac{d\boldsymbol{f}(\theta)}{d\theta} = G\boldsymbol{f}$ is $\boldsymbol{f}(\theta) = \exp(\theta G)\boldsymbol{f}(0)$. Now it is clear that if $\exists \theta_1$ such that $\boldsymbol{f}(\theta_1) = \boldsymbol{x}(\theta_1)$, then $\boldsymbol{f}(0) = \boldsymbol{x}(0)$, and therefore, $\boldsymbol{f}(\theta) = \exp(\theta G)\boldsymbol{x}(0) = \boldsymbol{x}(\theta), \forall \theta$.

Now for the case $k = 2$,

Let $y(\theta) = \frac{df(\theta)}{d\theta}$, then we have

$$\frac{d\boldsymbol{y}(\theta)}{d\theta} = G\boldsymbol{y}(\theta),$$

which solution is

$$\boldsymbol{y}(\theta) = \exp(\theta G)\boldsymbol{y}(0).$$

Thus, $\frac{d\boldsymbol{f}}{d\theta} = \exp(\theta G)\frac{df(\theta)}{d\theta}|_{\theta=0}$. Now since

$$\exp(\theta G)G\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} = \sum_{n \geqslant 0} \frac{G^n}{n!}G\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} = \sum_{n \geqslant 0} G\frac{G^n}{n!}\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} = G\exp(\theta G)\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0}$$

we have that,

$$\boldsymbol{f}(\theta) = \exp(\theta G)G^{-1}\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} + c\boldsymbol{1},$$

where $c \in \mathbb{R}$ and $\boldsymbol{1}$ denotes the d-dimensional vector of 1. Let's now add the interpolation condition, that is

$$\exists \theta_1, \theta_2, \text{ s.t. } \boldsymbol{f}(\theta_1) = \boldsymbol{x}(\theta_1), \ \ \boldsymbol{f}(\theta_2) = \boldsymbol{x}(\theta_2)$$

Which is equivalent to

$$\begin{cases} \exp(\theta_1 G)G^{-1}\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} + c\boldsymbol{1} = \exp(\theta_1 G)x(0) \\ \exp(\theta_2 G)G^{-1}\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} + c\boldsymbol{1} = \exp(\theta_2 G)x(0) \end{cases}$$

Which implies that, $\frac{d\boldsymbol{f}(\theta)}{d\theta}|_{\theta=0} = Gx(0)$ and that $c = 0$.

Therefore,

$$\boldsymbol{f}(\theta) = \exp(\theta G)x(0) = x(\theta), \forall \theta$$

$\blacksquare$

A.4. Proof of Proposition 2

Proof

$$\left\| \boldsymbol{D}(\theta) - (I + \sum_{k=1}^{h} \epsilon_k G_k)\boldsymbol{D}(\theta') \right\|_2^2 = \left\| \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta') - \sum_{k=1}^{h} \epsilon_k G_k \boldsymbol{D}(\theta') \right\|_2^2$$

$$= \left\langle \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta'), \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta') \right\rangle$$

$$- 2\left\langle \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta'), \sum_{k=1}^{h} \epsilon_k G_k \boldsymbol{D}(\theta') \right\rangle$$

$$+ \left\langle \sum_{k=1}^{h} \epsilon_k G_k \boldsymbol{D}(\theta'), \sum_{k=1}^{h} \epsilon_k G_k \boldsymbol{D}(\theta') \right\rangle,$$

Now, $\forall j \in \{1, \ldots, h\}$

$$\frac{\delta \left\| \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta') - \sum_{k=1}^{h} \epsilon_k G_k \boldsymbol{D}(\theta') \right\|_2^2}{\delta \epsilon_j} = -2(\boldsymbol{D}(\theta) - \boldsymbol{D}(\theta'))^T G_j \boldsymbol{D}(\theta') + 2\sum_{k=1}^{h} \epsilon_k \boldsymbol{D}(\theta')^T G_k^T G_j \boldsymbol{D}(\theta'),$$

setting $\frac{\delta\left\|\boldsymbol{D}(\theta)-\boldsymbol{D}(\theta')-\sum_{k=1}^{h}\epsilon_k G_k \boldsymbol{D}(\theta')\right\|_2^2}{\delta\epsilon_j} = 0$, for all j we obtain

$$\epsilon^* = \begin{pmatrix} \|G_1 \boldsymbol{D}(\theta')\|_2^2 & \dots & \boldsymbol{D}(\theta')^T G_h^T G_1 \boldsymbol{D}(\theta') \\ \vdots & \ddots & \vdots \\ \boldsymbol{D}(\theta')^T G_1^T G_h \boldsymbol{D}(\theta') & \dots & \|G_h \boldsymbol{D}(\theta')\|_2^2 \end{pmatrix}^{-1} \begin{pmatrix} (\boldsymbol{D}(\theta)-\boldsymbol{D}(\theta'))^T G_1 \boldsymbol{D}(\theta') \\ \vdots \\ (\boldsymbol{D}(\theta)-\boldsymbol{D}(\theta'))^T G_h \boldsymbol{D}(\theta') \end{pmatrix}$$

, and we have that

$$\begin{pmatrix} \|G_1 \boldsymbol{D}(\theta')\|_2^2 & \dots & \boldsymbol{D}(\theta')^T G_h^T G_1 \boldsymbol{D}(\theta') \\ \vdots & \ddots & \vdots \\ \boldsymbol{D}(\theta')^T G_1^T G_h \boldsymbol{D}(\theta') & \dots & \|G_h \boldsymbol{D}(\theta')\|_2^2 \end{pmatrix} = \begin{pmatrix} G_1 \boldsymbol{D}(\theta') \\ \vdots \\ G_h \boldsymbol{D}(\theta') \end{pmatrix}^T \begin{pmatrix} G_1 \boldsymbol{D}(\theta') \\ \vdots \\ G_h \boldsymbol{D}(\theta') \end{pmatrix}$$

which is thus a positive definite matrix. ∎

A.5. Proof of Proposition 3

Proof Given $J_{\omega'}[\boldsymbol{D}] = A_{\omega'}^D$ and $J_{\omega}[\boldsymbol{D}] = A_{\omega}^D$, we have

$$\left\| A_{\omega'}^D - A_{\omega}^D - \sum_{k=1}^{h} \epsilon_k G_k A_{\omega}^D \right\|_F^2 = \mathrm{Tr}\left((A_{\omega'}^D - A_{\omega}^D - \sum_{k=1}^{h} \epsilon_k G_k A_{\omega}^D) \odot (A_{\omega'}^D - A_{\omega}^D - \sum_{k=1}^{h} \epsilon_k G_k A_{\omega}^D)11^T \right)$$

$$= \mathrm{Tr}(A_{\omega'}^D \odot A_{\omega'}^D - A_{\omega'}^D \odot A_{\omega}^D - A_{\omega'}^D \odot (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D) + A_{\omega}^D \odot A_{\omega}^D$$

$$- A_{\omega}^D \odot A_{\omega'}^D + A_{\omega}^D \odot (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D) - (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D) \odot A_{\omega'}^D$$

$$+ (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D) \odot A_{\omega}^D + (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D) \odot (\sum_{h=1}^{k} \epsilon_k G_k A_{\omega}^D))11^T).$$

Now, $\forall j \in \{1, \dots, h\}$

$$\frac{\delta \left\| A_{\omega'}^D - A_{\omega}^D - \sum_{k=1}^{h} \epsilon_k G_k A_{\omega}^D \right\|_F^2}{\delta\epsilon_j} = 2\,\mathrm{Tr}\left((G_j A_{\omega}^D) \odot (A_{\omega}^D - A_{\omega'}^D + \sum_{k=1}^{h} G_k A_{\omega}^D 11^T) \right)$$

$$= 2\,\mathrm{Tr}\left(G_j A_{\omega}^D \odot (A_{\omega}^D - A_{\omega'}^D)11^T \right)$$

$$+ 2\sum_{k=1}^{h} \epsilon_k\,\mathrm{Tr}\left((G_j A_{\omega}^D \odot G_k A_{\omega}^D)11^T \right),$$

setting $\frac{\delta\left\| A_{\omega'}^D - A_{\omega}^D - \sum_{k=1}^{h} \epsilon_k G_k A_{\omega}^D \right\|_F^2}{\delta\epsilon_j} = 0$ for all j and rearranging in matrix form gives

$$\epsilon^* = \begin{pmatrix} \sum_i \|G_1[A_{\omega}^D]_{.,i}\|_2^2 & \dots & \sum_i \langle G_1[A_{\omega}^D]_{.,i}, G_h[A_{\omega}^D]_{.,i}\rangle \\ \vdots & \ddots & \vdots \\ \sum_i \langle G_h[A_{\omega}^D]_{.,i}, G_1[A_{\omega}^D]_{.,i}\rangle & \dots & \sum_i \|G_h[A_{\omega}^D]_{.,i}\|_2^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_i \langle G_1[A_{\omega}^D]_{.,i}, [A_{\omega'}^D]_{.,i} - [A_{\omega}^D]_{.,i}\rangle \\ \vdots \\ \sum_i \langle G_h[A_{\omega}^D]_{.,i}, [A_{\omega'}^D]_{.,i} - [A_{\omega}^D]_{.,i}\rangle \end{pmatrix},$$

and we have that

$$\begin{pmatrix} \sum_i \|G_1[A_{\omega}^D]_{.,i}\|_2^2 & \dots & \sum_i \langle G_1[A_{\omega}^D]_{.,i}, G_h[A_{\omega}^D]_{.,i}\rangle \\ \vdots & \ddots & \vdots \\ \sum_i \langle G_h[A_{\omega}^D]_{.,i}, G_1[A_{\omega}^D]_{.,i}\rangle & \dots & \sum_i \|G_h[A_{\omega}^D]_{.,i}\|_2^2 \end{pmatrix} = \sum_{i=1}^{h} \begin{bmatrix} G_1[A_{\omega}^D]_{.,i} \\ \vdots \\ G_h[A_{\omega}^D]_{.,i} \end{bmatrix}^T \begin{bmatrix} G_1[A_{\omega}^D]_{.,i} \\ \vdots \\ G_h[A_{\omega}^D]_{.,i} \end{bmatrix},$$

therefore it is the sum of positive definite matrices.

For the case h=1, we have that

$$\left\| a_{\omega'}^D - a_\omega^D - \epsilon G a_\omega^D \right\|^2 = \left\langle a_{\omega'}^D, a_{\omega'}^D \right\rangle - 2 \left\langle a_\omega^D, a_{\omega'}^D \right\rangle + \left\langle a_\omega^D, a_\omega^D \right\rangle$$
$$+ 2 \left\langle \epsilon G a_\omega^D, a_\omega^D - a_{\omega'}^D \right\rangle + \left\langle \epsilon G a_\omega^D, \epsilon G a_\omega^D \right\rangle,$$

thus,

$$\frac{\delta \left\| a_{\omega'}^D - a_\omega^D - \epsilon G a_\omega^D \right\|^2}{\delta \epsilon} = a_\omega^{D^T} G^T (a_\omega^D - a_{\omega'}^D) + \epsilon a_\omega^{D^T} G^T G a_\omega^D$$

■

For the following proofs, we will denote by $T : \mathbb{R}^d \times \mathbb{R}^h \to \mathbb{R}^d$, the transformation operator taking as input a datum and a group parameter, and giving as output the transformed datum. As we used a Lie group, we can define this operator analytically as $T(x, \theta) = \exp(\theta G) x$.

A.6. Proof of Theorem 2

For this proof, we will use the notation $\mathcal{T}_{\mathcal{X}}(\omega)$ as the tangent space of the manifold described by the data $\mathcal{X}$ for the data in the region ω, and by $\mathcal{T}_{AE}(\omega)$ the tangent space of the AE for the region ω. We show that if these two tangent space coincides for a given region, i.e., if the tangent space of the AE coincides with the tangent space of the manifold for a specific position, then they coincide everywhere.

Proof By assumption, we know that $\{a_1^D(\omega'), \ldots, a_h^D[\omega']\}$ form a basis of $\mathcal{T}_{\mathcal{X}}[\omega']$. If the regularization is satisfied, we also know that the tangent induced by the AE at position ω, denoted by $\mathcal{T}_{AE}(\omega)$, is equal to $T(\mathcal{T}_{\mathcal{X}}(\omega'), \theta)$. In fact, for the order $k = 2$ the regularization imposes that the tangent (induced by the AE) of the different regions are transformed version of each other by the transformation operator T. Now for the order one, we know that if $\frac{d\boldsymbol{f}(\theta)}{d\theta} = G\boldsymbol{f}(\theta)$, then $\frac{d^2 \boldsymbol{f}(\theta)}{d\theta^2} = G \frac{d\boldsymbol{f}(\theta)}{d\theta}$. Which means that if the outputs of the interpolant $\boldsymbol{f}$ are connected by the transformation group T, then the tangents of such interpolant are also connected by the same group of transformation.

Note that the operator T forms a Lie group action operator, it is a diffeomorphism from the orbit of the group to the orbit of the group. Therefore, $\forall \omega$, it exists θ such that $T(\mathcal{T}_{\mathcal{X}}(\omega'), \theta) = \mathcal{T}_{\mathcal{X}}(\omega)$. Per assumption, the tangent of the region ω', i.e. $\mathcal{T}_{AE}(\omega')$ is actually tangent to $\mathcal{X}$ as its basis coincides with $\mathcal{T}_{\mathcal{X}}(\omega')$. Denote by $x \in \mathcal{X}$ the point at which $\mathcal{T}_{\mathcal{X}}(\omega')$ and $\mathcal{X}$ intersects. Let's first first prove that for $\epsilon' = \arg\max_\epsilon x + \epsilon h \in \omega$, where $h \in \mathcal{T}_{\mathcal{X}}(\omega')$, that is, $x + \epsilon' h$ lies at the boundary of the region ω'. We further assume that $\|h\| = 1$ such that $\epsilon' = \text{Rad}(\omega')$. Let's define a smooth curve on the manifold $\gamma : \mathbb{R} \to \mathcal{X}$ such that $\gamma(0) = x$ and $\gamma'(0) = h$. Now,

$$d(x + \epsilon' h, \mathcal{X}) \leqslant d(x + \epsilon' h, \gamma(\epsilon'))$$
$$= \left\| \gamma(\epsilon') - \gamma(0) - \epsilon' \gamma'(0) \right\|.$$

Since, $\lim_{\epsilon' \to 0} \frac{\gamma(\epsilon') - \gamma(0)}{\epsilon'} = \gamma'(0)$, we have that $\frac{d(x + \epsilon' h, \gamma(\epsilon'))}{\epsilon'} = o(\text{Rad}(\omega'))$. Then, since the $\omega_i \forall i \in \{1, \ldots, |\Omega|\}$ form a partition of Ω and that by Proposition 2 we know that since one tangent of the AE coincides with the tangent of the manifold at the point x then any tangent of the AE coincides with a tangent of the manifold. Thus, we have that $d\left(\cup_{\omega \in \Omega} \mathcal{T}_{AE}(\omega), \mathcal{X} \right) = \sum_{i=1}^{|\Omega|} d\left(\mathcal{T}_{AE}(\omega_i), \mathcal{X} \right) \leqslant \sum_{i=1}^{|\Omega|} \text{Rad}(\omega_i)$.

■

A.7. Per Region Tangent - Details

Let $[\mathbf{D} \circ \mathbf{E}(.)]_i : \mathbb{R}^d \to \mathbb{R}$ be the ith coordinate output of the AE, defined as $[\mathbf{D} \circ \mathbf{E}(x)]_i = [A_\omega^D]_{i,.} A_\omega^E x + [A_\omega^D]_{i,.} B_\omega^E + [B_\omega^D]_i$.

$$d[\mathbf{D} \circ \mathbf{E}(.)]_i = [\mathbf{D} \circ \mathbf{E}(x + \epsilon)]_i - [(\mathbf{D} \circ \mathbf{E})(x)]_i = \left\langle A_\omega^{E^T}[A_\omega^D]_{i,.}^T, \epsilon \right\rangle, \forall \epsilon \in \mathbb{R}^d. \tag{14}$$

As such, we directly obtain that

$$\nabla_x[\mathbf{D} \circ \mathbf{E}(.)]_i = A_\omega^{E^T}[A_\omega^D]_{i,.}^T, \tag{15}$$

which leads to the Jacobian of the AE as defined in Eq. 7.

A.8. Higher-order Contractive Autoencoder: Details

Let consider the case of a 1 hidden-layer encoder, follows by any depth encoder. In the second order regularization, one penalizes $\left\| A_\omega^E - A_{\omega'}^E \right\|_F$, where ω and ω' are neighboring regions. We know that $A_\omega^E = Q_\omega^1 W^1$, now let consider the case of a 3 ReLU-units encoder, that is, Q_ω^1 is a 3×3 diagonal matrix, and $W^1 \in \mathbb{R}^{3 \times n}$, where n is the input space dimension. A particular case we consider for our analysis is, $Q_\omega^1 = \text{Diag}(1, 0, 1)$, and $Q_{\omega'}^1 = \text{Diag}(1, 1, 1)$, i.e., the first region ω is encoded by 2 activated ReLUs and ω' by 3. The associated HOC penalization is $\left\| \begin{pmatrix} W_{1,:}^1 \\ 0 \\ W_{3,:}^1 \end{pmatrix} - \begin{pmatrix} W_{1,:}^1 \\ W_{2,:}^1 \\ W_{3,:}^1 \end{pmatrix} \right\|_F = \left\| W_{2,:}^1 \right\|_2$, where $W^1 = \begin{pmatrix} W_{1,:}^1 \\ W_{2,:}^1 \\ W_{3,:}^1 \end{pmatrix}$.

Therefore we see even if $W_{1,:}^1$ or $W_{3,:}^1$ are large, they will not induce a penalization of the curvature between the region ω and ω'. Besides, if $W_{2,:}^1$ is small, even thought it is associated with the changing unit between the two regions, the curvature will not be penalized either.

Appendix B. Optimal Parameters

The following proposition provides the optimal transformation parameters $\epsilon^\star = [\epsilon_1, \ldots, \epsilon_h]^T$ that are used during the learning of the regularized AE.

Proposition 2 *The ϵ of the first-order regularization defined in Eq. 12 is obtained as*

$$\epsilon^* = \begin{pmatrix} \|G_1 \boldsymbol{D}(\theta')\|_2^2 & \ldots & \langle G_h \boldsymbol{D}(\theta'), G_1 \boldsymbol{D}(\theta') \rangle \\ \vdots & \ddots & \vdots \\ \langle G_1 \boldsymbol{D}(\theta'), G_h \boldsymbol{D}(\theta') \rangle & \ldots & \|G_h \boldsymbol{D}(\theta')\|_2^2 \end{pmatrix}^{-1} \begin{pmatrix} \langle \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta'), G_1 \boldsymbol{D}(\theta') \rangle \\ \vdots \\ \langle \boldsymbol{D}(\theta) - \boldsymbol{D}(\theta'), G_h \boldsymbol{D}(\theta') \rangle \end{pmatrix}$$

where the matrix is always invertible ($\boldsymbol{D}(\theta') \neq 0$). (Proof in Appendix A.4.)

Proposition 3 *The ϵ of the second-order regularization defined in Eq. 13 is obtained by*

$$\epsilon^* = \begin{pmatrix} \Sigma_i \|G_1[A_\omega^D]_{.,i}\|_2^2 & \ldots & \Sigma_i \langle G_1[A_\omega^D]_{.,i}, G_h[A_\omega^D]_{.,i} \rangle \\ \vdots & \ddots & \vdots \\ \Sigma_i \langle G_h[A_\omega^D]_{.,i}, G_1[A_\omega^D]_{.,i} \rangle & \ldots & \Sigma_i \|G_h[A_\omega^D]_{.,i}\|_2^2 \end{pmatrix}^{-1} \begin{pmatrix} \Sigma_i \langle G_1[A_\omega^D]_{.,i}, [A_{\omega'}^D]_{.,i} - [A_\omega^D]_{.,i} \rangle \\ \vdots \\ \Sigma_i \langle G_h[A_\omega^D]_{.,i}, [A_{\omega'}^D]_{.,i} - [A_\omega^D]_{.,i} \rangle \end{pmatrix},$$

where the matrix is invertible ($A_\omega^D \neq 0$). (Proof in Appendix A.5.)

Appendix C. Orbit of a Lie Group

One example of the orbit of a data with respect to a Lie group is the result of the rotation on an initial point $x(0) \in \mathbb{R}^2$, we have $x(\theta) = \exp(\theta G)x(0), \theta \in \mathbb{R}, G = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$. In fact, where we recall that $\exp\left(\theta \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\right) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$. The infinitesimal operator G is thus encapsulating the group information. For more details regarding Lie group and the exponential map refer to Hall (2015).

Appendix D. Datasets

Most of the datasets used for the experiments are extracted from the univariate time-series repository in Bagnall et al. (2017). Some of them are recording from sensors or simulated data. They range from motion time-series to biological one. The dimension of the data we used is between $200 - 1000$.

MNIST (Deng, 2012), is a handwritten digit dataset containing 60.000 training and 10.000 test images of dimension 28×28 representing 10 classes. This is an example of a dataset where the intra-class viariability is induced by group transformations such as rotation, translation, and small diffeomorphism

CIFAR10 (Krizhevsky, 2009), consists of 50.000 training colour images and 10.000 test color images of dimension 32×32 representing 10 classes. In this dataset, the transformation required to go from one data to another requires more than simple rigid transformations.

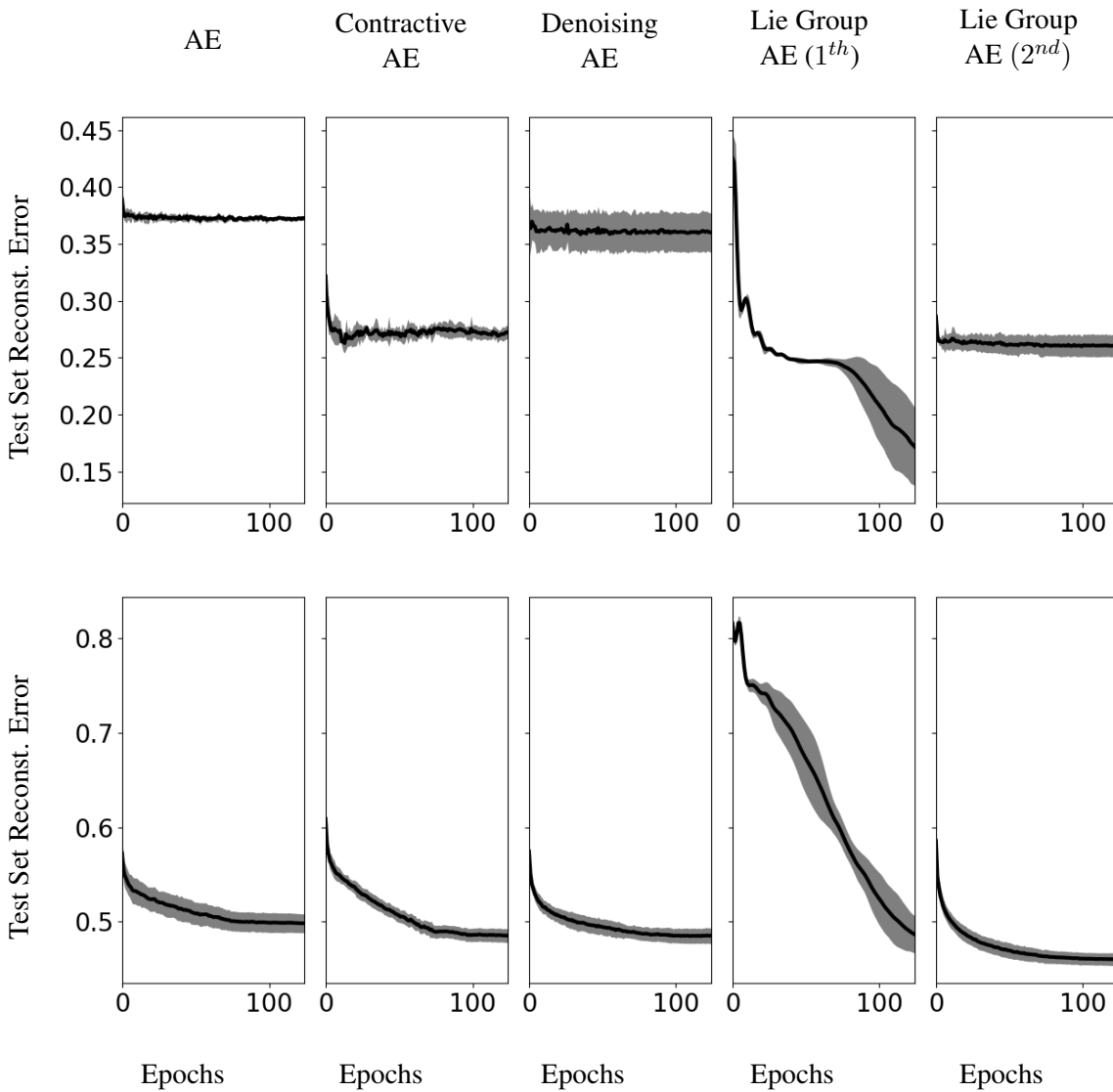## Appendix E. Additional Experimental Figures

Figure 6: Test set reconstruction error on the (*top*) Haptics and (*bottom*) Facefour datasets evaluated on the best set of parameters for different AEs (from left to right): AE, Higher Order Contractive AE, Denoising AE, Lie Group AE (first-order), and Lie Group AE (second-order). For each model, the mean over 10 runs is reported in black, and the gray area corresponds to its standard deviation.