FlexAE: Flexibly Learning Latent Priors for Wasserstein Auto Encoders Supplementary Material

Arnab Kumar Mondal[1] **Himanshu Asnani**[2] **Parag Singla**[1] **Prathosh AP**[1]

[1]IIT, Delhi, India
[2]TIFR, Mumbai, India

Abstract

Auto-Encoder (AE) based neural generative frameworks model the joint-distribution between the data and the latent space using an Encoder-Decoder pair, with regularization imposed in terms of a prior over the latent space. Despite their advantages, such as stability in training, efficient inference, the performance of AE based models has not reached the superior standards of the other generative models such as Generative Adversarial Networks (GANs). Motivated by this, we examine the effect of the latent prior on the generation quality of deterministic AE models in this paper. Specifically, we consider the class of Generative AE models with deterministic Encoder-Decoder pair (such as Wasserstein Auto-Encoder (WAE), Adversarial Auto-Encoder (AAE)), and show that having a fixed prior distribution, *a priori*, oblivious to the dimensionality of the 'true' latent space, will lead to the infeasibility of the optimization problem considered. As a remedy to the issue mentioned above, we introduce an additional state space in the form of flexibly learnable latent priors, in the optimization objective of WAE/AAE. Additionally, we employ a latent-space interpolation based smoothing scheme to address the non-smoothness that may arise from highly flexible priors. We show the efficacy of our proposed models, called FlexAE and FlexAE-SR, through several experiments on multiple datasets, and demonstrate that FlexAE-SR is the new state-of-the-art for the AE based generative models in terms of generation quality as measured by several metrics such as Fréchet Inception Distance, Precision/Recall score. Code for our paper is available at: `https://github.com/dair-iitd/FlexAE`

1 INTRODUCTION

Auto-Encoder (AE) based latent variable generative models implicitly define a joint distribution over the input data and a lower-dimensional latent space, by approximating the true latent posterior, with a variational distribution. This variational distibution is parameterized using a neural network called the Encoder. The distribution induced by the Encoder is regularized to follow a pre-defined latent prior distribution. Subsequently, a Decoder network is trained to conditionally sample from the data distribution via optimizing a data-reconstruction metric. The parameters of the Encoder and the Decoder networks are learnt by optimizing either a bound on the data likelihood [Kingma and Welling, 2014] or a divergence measure between the true and generated data distributions [Tolstikhin et al., 2018]. The framework of AE-based generative models is attractive because of its ease and stability in training, efficiency in sampling, and flexibility in architectural choices. However, despite their advantages, AE-based models have failed to reach the performance of other State-of-The-Art (SoTA) generative models [Dai and Wipf, 2019, Mondal et al., 2020].

Several aspects such as the loss function used for optimization [Higgins et al., 2017, Larsen et al., 2016], presence of conflicting terms in the optimization objective [Hoffman and Johnson, 2016, Kim and Mnih, 2018], distributional choices (e.g., Gaussianity) imposed on the Encoder and Decoder [Zhao et al., 2019, Rezende and Viola, 2018], dimensionality of the latent space used [Dai and Wipf, 2019, Mondal et al., 2020], the mismatch between the learned and imposed prior [Shengjia Zhao and Ermon, 2017, Tomczak and Welling, 2018] have been identified as possible causes for the sub-optimal performance of the AE-based models. Many remedial measures, including the modification of the objective function [Zhao et al., 2019, Higgins et al., 2017, Kim and Mnih, 2018], use of non-Gaussian Encoder/Decoder [Larsen et al., 2016, Nalisnick et al., 2016], masking of spurious latent dimensions [Mondal et al., 2020], incorporating a richer class of priors on the latent space [Tomczak and

Welling, 2018, Takahashi et al., 2019, Klushyn et al., 2019], have been proposed in the literature to address some of these issues. While these modifications have improved AE models' performance, they are still behind SoTA generative models [Dai and Wipf, 2019, Mondal et al., 2020]. In this work, we make the following contributions:

1. We theoretically establish that in a deterministic AE based generative model (such Wasserstein AE), choosing a latent prior distribution supported on the entire space, leads to an infeasible optimization objective, when the model's latent space has dimensionality that is other than that of the 'true' latent space.
2. As a remedy, we propose a new model called FlexAE, that can impose flexible learnable priors on WAEs to make the optimization problem feasible by introducing an additional state space over the latent prior.
3. We employ an adversarial regularization technique to smooth the latent space of our model with flexible priors to prevent memorization. Furthermore, We propose two novel metrics, called, Pixel Memorization Score and Inception Memorization Score, to quantitatively evaluate whether the generative AE has memorized the training samples.
4. We empirically demonstrate our claims through extensive experimentation on synthetic and real-world datasets by achieving significant improvement in generation quality over the SoTA AE-based generative models.

2 BACKGROUND

The general theme in majority of the AE based generative frameworks is to implicitly learn the joint distribution between the observed data and a latent variable, via optimizing an objective function which consists of an autoencoding (conditional likelihood) and a latent regularization term (divergence measure). Variational Autoencoder (VAE) [Kingma and Welling, 2014] is the pioneering member of this family, in which the variational latent posterior and conditional data likelihood are respectively parameterized by probabilistic (Gaussian) Encoder and Decoder networks, while the latent prior is assumed to be an isotropic Gaussian distribution. A related class of AE-models are the Adversarial Auto-Encoders (AAEs) [Makhzani et al., 2016] and Wasserstein Auto-Encoders (WAEs) [Tolstikhin et al., 2018] where a pair of deterministic Encoder-Decoder is used with Jensen-Shannon and Wasserstein distance respectively, between the aggregated latent posterior and the latent prior.

Although VAEs/WAEs provide solid frameworks for AE-based generative models, several drawbacks are associated with them, which prevents them to compete with the other SoTA generative models. It is shown that there exists a conflict between the two terms of the objective, in the case of VAEs [Higgins et al., 2017, Shengjia Zhao and Ermon, 2017, Rezende and Viola, 2018]. A few remedial measures such as introduction of a tunable parameter in the objective [Burgess et al., 2017], use of additional penalties such as mutual information [Zhao et al., 2019], total correlation [Kim and Mnih, 2018], and generalised optimization objective [Rezende and Viola, 2018] have been proposed. Another often discussed issue with AE-models with stochastic Encoder-Decoders is that they adopt a simple unimodal Gaussian distribution for parameterization [Rosca et al., 2018]. To address this, [Nalisnick and Smyth, 2017] implements a Bayesian non-parametric version of the variational autoencoder that has a latent representation with stochastic dimensionality and could represent richer class of distributions. Invertible flow-based generative models [Kingma et al., 2016, Rezende and Mohamed, 2015] capitalize on the idea of normalizing flow for the Encoder and Decoder networks. VAE-GAN [Larsen et al., 2016], VGH/VGH++ [Rosca et al., 2018] incorporates an adversarial learning at the Decoder so that it can represent a rich class of distributions.

Further, it is observed that there is a mismatch between the aggregated variational posterior and the latent prior, leading to sub-optimality of the divergence term in the objective and in turn poor generation [Tomczak and Welling, 2018, Dai and Wipf, 2019]. Several methods try to alleviate this problem, broadly in two ways (i) using a richer class of parametric priors on the latent space [Tomczak and Welling, 2018, Klushyn et al., 2019, Kumar et al., 2020] and (ii) using a post-hoc technique to minimize the divergence or sample from the latent space without regularizing it [Bauer and Mnih, 2019, Ghosh et al., 2020, Takahashi et al., 2019]. Among the first category of methods, VampPrior [Tomczak and Welling, 2018] assumes the prior to be a mixture of the conditional posteriors with a set of learnable pseudo-inputs. Klushyn et al. [2019] adapt the constrained optimization setting in [Rezende and Viola, 2018] and substitute the standard normal prior with a hierarchical prior and use an importance-weighted bound as the optimization objective. Huang et al. [2017], Kumar et al. [2020] learn the latent priors using normalizing flow based methods. Within the second category of methods, Bauer and Mnih [2019] learn to sample from a rich class of priors by multiplying a simplistic prior distribution with a learned acceptance function. Takahashi et al. [2019] used kernel density trick for matching the prior to the aggregated posterior. RAE-GMM [Ghosh et al., 2020], imposes an L2-norm penalty in the latent space and learns to sample from it using a Gaussian Mixture Model (GMM) on the latent space. While these methods report improvement over the SoTA metrics, not many give a theoretical justification for using richer-class of latent priors. Further, post-hoc latent samplers such as RAE-GMM do not have control over the amount of bias imposed (other than a simple objective scaling factor), that might lead to over/under fitting.

However, it has been both theoretically and empirically observed that dimensionality of the latent space used has a critical impact on the performance of these models [Mondal et al., 2020, Dai and Wipf, 2019, Rubenstein et al., 2018]. Dai and Wipf [2019] study the implication of the mismatch between the dimensionality of the data and the true latent space and the role of Decoder variance, in the case of AEs with stochastic Encoders. They argue a learnable variance in the Decoder would make the objective reach negative infinity even when the aggregated posterior would not match the standard Gaussian prior not because of simplistic modelling assumption but because of mismatch between data dimensionality and the true latent dimensionality. To resolve this issue they introduce a second-stage VAE, which is used on the latent space of the first stage (which is a usual VAE), where the data and the latent dimensions match. In MaskAAE [Mondal et al., 2020], the authors noted that the generation quality degrades when there is a mismatch between the dimensionality of the true and the assumed latent space of a deterministic AE. They develop a procedure to explicitly zero-out (mask) the spurious latent dimensions via a learnable masking layer. In this backdrop, to the best of our knowledge, ours is the first study to propose a flexibly learnable prior scheme on deterministic AEs as a solution to the problem of infeasibility of the objective function.

3 PROPOSED METHOD

3.1 OPTIMALITY OF THE LATENT SPACE OF WASSERSTEIN AE

We start by assuming that the true data is generated in nature via a two-step process. First, the true latent variables are sampled from an n-dimensional space, $\widetilde{\mathcal{Z}}$ according to some continuous distribution in $\mathbb{R}^n$. Next, a non-linear function, $f : \widetilde{\mathcal{Z}} \rightarrow \mathcal{X}$ maps the true latent space, $\widetilde{\mathcal{Z}}$ to the observed data space, $\mathcal{X} \subseteq \mathbb{R}^d$, with $d >> n$, in most practical cases. In other words, observed data $\boldsymbol{x}$ lies on $\mathcal{X}$, an n-dimensional manifold embedded in $\mathbb{R}^d$. We make a benign assumption on f that it can be represented using neural networks with sigmoidal (or hyperbolic tangent, ReLU, Leaky ReLU etc.) activations to arbitrary closeness. Under this model, the data could be seen as lying in an n-dimensional manifold within $\mathbb{R}^d$, with an underlying ground truth distribution $P_d(\boldsymbol{x})$. The objective of a Generative AE such as WAE [Tolstikhin et al., 2018] or AAE[1] [Makhzani et al., 2016] is to estimate (or learn to sample from) the distribution $P_d(\boldsymbol{x})$, given some i.i.d. samples drawn from it. The distribution learned by the model, denoted by $P_\theta(\boldsymbol{x})$ is given by $P_\theta(\boldsymbol{x}) = \int_{\mathcal{Z}} P_\theta(\boldsymbol{x}|\boldsymbol{z}) dP_z$, where $P_\theta(\boldsymbol{x}|\boldsymbol{z})$ is the distribution parameterized by a deterministic Decoder neural network $D_\theta(\boldsymbol{z})$ and $P_Z(\boldsymbol{z})$ is the latent prior defined on an m-dimensional space, $\mathcal{Z}$. WAEs learn to sample from the distribution $P_\theta(\boldsymbol{x})$ by solving the following optimization problem:

$$\inf_{\phi,\theta} \left(\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})} \mathop{\mathbb{E}}_{Q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[c\Big(\boldsymbol{x}, D_\theta\big(E_\phi(\boldsymbol{x})\big)\Big) \right] \right) \quad \textit{such that } Q_\phi(\boldsymbol{z}) = P_Z(\boldsymbol{z}) \tag{1}$$

Here $Q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is the variational conditional posterior, which is also parameterized by a neural network called the Encoder, $E_\phi : \mathcal{X} \rightarrow \mathcal{Z}$. When E_ϕ is deterministic, $Q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is dirac-delta for every $\boldsymbol{x}$. $Q_\phi(z) = \int_{\mathbb{R}^d} Q_\phi(\boldsymbol{z}|\boldsymbol{x})\, dP_d(\boldsymbol{x})$ is the aggregated posterior distribution imposed by the Encoder, $c : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$ is any measurable cost function (such as Mean Square Error (MSE), Mean Absolute Error (MAE) or adversarial loss and so on) and $\phi \in \Phi$, $\theta \in \Theta$ are the learnable parameters of Encoder and Decoder, respectively. The constrained optimization problem in Eq. 1 translates to auto-encoding the input data with a constraint (regularizer) that the aggregated distribution imposed by the Encoder matches with a predefined latent prior distribution. The constrained optimization objective of WAE (Eq. 1) can be equivalently written as an unconstrained problem by introducing a Lagrangian:

$$D_{WAE} = \inf_{\phi,\theta} \left(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})} \mathop{\mathbb{E}}_{Q_\phi(\boldsymbol{z}|\boldsymbol{x})} \left[c\Big(\boldsymbol{x}, D_\theta\big(E_\phi(\boldsymbol{x})\big)\Big) \right]}_{\text{a}} + \lambda \cdot \underbrace{D_Z\big(Q_\phi(\boldsymbol{z}), P_Z(\boldsymbol{z})\big)}_{\text{b}} \right) \tag{2}$$

Where λ is the Lagrange multiplier, $D_Z(.)$ is any divergence measure such as Kullaback-Leibler, Jenson-Shannon or Wasserstein distance, between two distributions.

Note that objective in Eq. 1 becomes feasible only when $D_Z\big(Q_\phi(\boldsymbol{z}), P_Z(\boldsymbol{z})\big)$ becomes zero. Equipped with these, in Theorem 1, we show that when $m > n$ (most common practical case), the optimization objective (Eq. 1) does not have a feasible solution when the prior is fixed a priori to be any distribution which is supported outside of a set of countable union of all possible n-dimensional manifolds in an m-dimensional space, denoted by $\mathcal{Q}_m^n$. An example for such a prior is the isotropic Gaussian distribution in $\mathbb{R}^m$, which is the usual choice in most models.

Theorem 1. *If $m > n$, then the regularization term in the objective function of an AE-based generative model (Eq. 2), cannot be driven to zero. That is $D_Z(Q_\phi(\boldsymbol{z}), P_Z(\boldsymbol{z})) > 0$, $\forall \phi$ and for any distributional divergence D_Z when the support of $P_Z(\boldsymbol{z}) \notin \mathcal{Q}_m^n$.*

The above theorem (cf. supp. for proof) asserts that it is impossible to match the aggregated latent posterior to the prior when the assumed latent dimension is more than the true latent dimension and the assumed prior has full-support. Even though a stochastic Encoder can fill the 'extra' dimensions with external noise, we only consider the case of

[1]AAE is a special case of WAE [Bousquet et al., 2017].

deterministic AE in this work, since stochasticity can lead to problems such as conflicting objectives [Burgess et al., 2017] and non-unique solutions [Dai and Wipf, 2019].

3.2 FLEXIBLY LEARNING PRIOR: FLEXAE

Based on the discussion so far, fixing a prior makes the optimization objective infeasible and no prior leads to overfitting. To alleviate these, we propose to flexibly learn the latent prior jointly with the AE-training by introducing an additional state-space in the original optimization objective (Equation 2) as follows:

$$D_{FlexAE} = \inf_{\psi,\phi,\theta} \Bigg(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})} \mathop{\mathbb{E}}_{Q(\boldsymbol{z}|\boldsymbol{x})} \Big[c\big(\boldsymbol{x}, D_\theta\big(E_\phi(\boldsymbol{x})\big)\big) \Big]}_{\text{a}} + \underbrace{\lambda \cdot D_Z\big(Q_\phi(\boldsymbol{z}) || P_\psi(\boldsymbol{z})\big)}_{\text{b}} \Bigg) \tag{3}$$

where $P_\psi(\boldsymbol{z})$ is a learnable latent prior parmaterized using a neural network called the Prior-Generator (P-GEN), G_ψ, that takes an $m' \geq n$ dimensional isotropic Gaussian distribution as the input and generates sample from an m-dimensional $P_\psi(\boldsymbol{z})$ (refer Fig. 1). In our model, referred to as the Flexible AE or FlexAE, P-GEN is jointly trained with the AE to alternatively minimize the divergence measure and the reconstruction terms in Eq. 3. Consequently, the output of the P-GEN forms the prior that is imposed on the latent space. Please note that $D_{FlexAE} \leq D_{WAE}$ and thus the new formulation does not harm the optimization. Next, the Corollary 1.1 (proof in the supp.) below states that the divergence measure can be brought to zero with FlexAE.

Corollary 1.1. *$\forall m' \geq n$, $D_Z(Q_\phi(\boldsymbol{z})||P_\psi(\boldsymbol{z}))$ (term (b) in FlexAE objective (Eq. 3) becomes zero for optimum set of parameters.*

3.2.1 Smoothing the Latent Space

It is well known that, unlike in AE models with stochastic Encoders (such as VAE), the latent space of a deterministic AEs tend to be dirac-deltas since there is no source of stochasticity beyond that is inherently present with the data [Rezende and Viola, 2018]. Consequently, a parametric prior generator network (as with FlexAE) may also 'memorize' the latent codes and eventually learn to generate samples very similar to those present in the training data. This defeats the purpose of a generative AE. To address this issue, we employ an adversarial smoothing regularization scheme along with the flexibly learnable prior. We adapt the adversarial regularizer proposed by Berthelot et al. [2019] to learn a smooth latent space.

Specifically, while training, first a convex combination of the encoded representations, $\boldsymbol{z}^{(j)}, \boldsymbol{z}^{(k)}$ of two randomly sampled training examples, $\boldsymbol{x}^{(j)}, \boldsymbol{x}^{(k)}$ are taken and the resulting vector is called interpolated latent, $\boldsymbol{z}_{inp}$, i.e. $z_{inp}^{(i)} = \gamma z^{(j)} + (1-\gamma) z^{(k)}$, where $\gamma \sim \mathcal{U}[0,1]$. Next, the decoder is trained to generate a realistic image, $\boldsymbol{x}_{inp} \sim P_\theta^c(\boldsymbol{x}_{inp})$ using $\boldsymbol{z}_{inp}$. Here, P_θ^c represents the distribution of images generated by decoding interpolated latent codes. Mathematically, we propose to minimize a divergence metric between the true data and the data generated by the decoder using the interpolated latent space. This imposes a local smoothness on the latent space since the decoder is forced to learn to generate not only from encoded latent codes but also from its random interpolations. We call the FlexAE model with smoothing regularization as FlexAE-SR. The objective in Equation 3 is modified as follows:

$$D_{FlexAE-SR} = \inf_{\psi,\phi,\theta} \Bigg(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})} \mathop{\mathbb{E}}_{Q(\boldsymbol{z}|\boldsymbol{x})} \Big[c\big(\boldsymbol{x}, D_\theta(\boldsymbol{z})\big) \Big]}_{\text{a}} + \lambda_1 \cdot \underbrace{D_Z\big(Q_\phi(\boldsymbol{z}) || P_\psi(\boldsymbol{z})\big)}_{\text{b}} + \lambda_2 \cdot \underbrace{D_X\big(P_\theta^c(\boldsymbol{x}_{inp}) || P_d(\boldsymbol{x})\big)}_{\text{c}} \Bigg) \tag{4}$$

where, $D_X(.)$ is any divergence measure such as Kullaback-Leibler, Jenson-Shannon or Wasserstein distance, between the two distributions, $P_\theta(\boldsymbol{x}_{inp})$ and $P_d(\boldsymbol{x})$. Similar interpolation technique is proven to be useful for learning better representation [Verma et al., 2019] and for enhancing the performance of generative model in unsupervised few-shot image generation [Wertheimer et al., 2020].

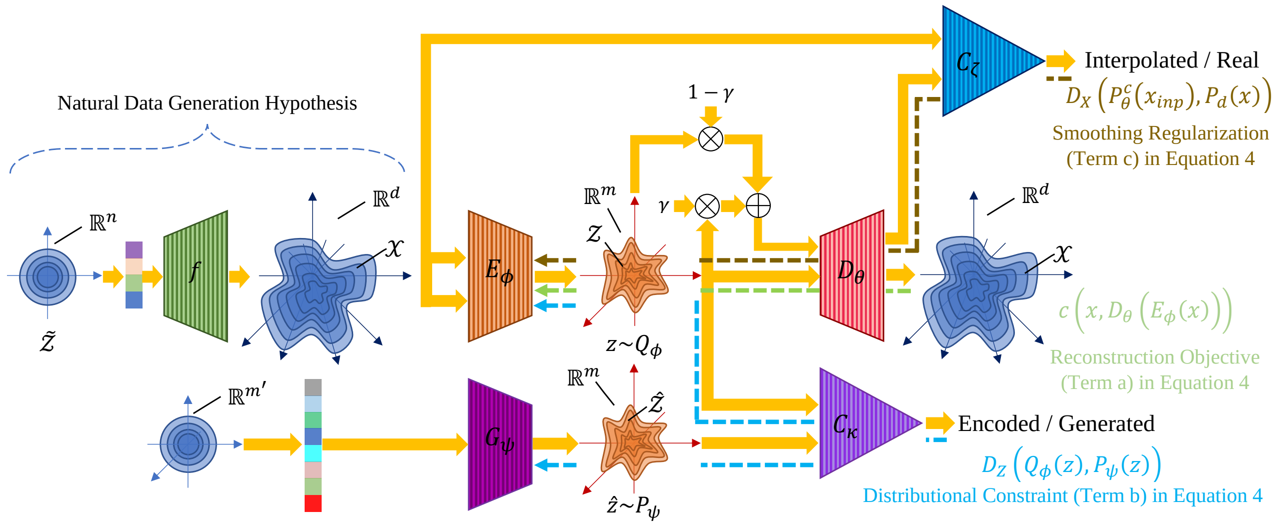
Please note, good generation may happen even when the divergence measure D_Z is not exactly zero but $Q_\phi(\boldsymbol{z})$ is 'close' enough to $P_Z(\boldsymbol{z})$. However, the closer $Q_\phi(\boldsymbol{z})$ and $P_Z(\boldsymbol{z})$ are, the better is the generation (see Figure 2 and 3), which is our claim (Theorem 1 is a singularity towards this).

3.2.2 Implementation

For implementation, we use MSE for c in term (a) of Eq. 4. D_Z, in principle can be chosen to be any distributional divergence such as Kullback-Leibler divergence (KLD), Jensen–Shannon divergence (JSD), Wasserstein Distance and so on. In this work, we propose to use Wasserstein distance and utilize the principle laid in [Arjovsky et al., 2017, Gulrajani et al., 2017], to optimize the divergence measure (term (b) in Equation 4). The loss functions used for different blocks of FlexAE are as follows:

1. Likelihood Loss - Realization of term a in Eq. 4:

$$L_{AE} = \frac{1}{s} \sum_{i=1}^{s} ||\boldsymbol{x}^{(i)} - D_\theta(E_\phi(\boldsymbol{x}^{(i)}))||^2 \tag{5}$$



Natural Data Generation Hypothesis
$\mathbb{R}^n$
$\tilde{\mathcal{Z}}$
f
$\mathbb{R}^d$
$\mathcal{X}$
$\mathbb{R}^{m'}$
E_ϕ
G_ψ
$\mathbb{R}^m$
$\mathcal{Z}$
$z \sim Q_\phi$
$\hat{\mathcal{Z}}$
$\hat{z} \sim P_\psi$
$1-\gamma$
γ
D_θ
C_ζ
Interpolated / Real
$D_X\left(P_\theta^c(x_{inp}), P_d(x)\right)$
Smoothing Regularization (Term c) in Equation 4
$c\left(x, D_\theta\left(E_\phi(x)\right)\right)$
Reconstruction Objective (Term a) in Equation 4
C_κ
Encoded / Generated
$D_Z\left(Q_\phi(z), P_\psi(z)\right)$
Distributional Constraint (Term b) in Equation 4

Figure 1: Proposed Model, FlexAE: Nature first samples an n-dimensional latent code from the true latent space, $\tilde{\mathcal{Z}}$. Next, the latent code is mapped to an n-dimensional manifold, $\mathcal{X}$ in $\mathbb{R}^d$. The observed variables are encoded using a deterministic Encoder, E_ϕ. The m-dimensional encoded representations lie in an n-dimensional manifold $\mathcal{Z}$. The decoder network, D_θ, learns an inverse projection from the learnt latent space, $\mathcal{Z}$ to the dataspace, $\mathcal{X}$. The generator network, G_ψ parameterizes the learnable prior distribution, that maps an isotropic Gaussian distribution in $\mathbb{R}^{m'}$ to any arbitrary prior $P_\psi(\boldsymbol{z})$ in $\mathbb{R}^m$. The critic network, C_κ measures the distributional divergence between Q_ϕ and P_ψ. The smoothness regulator, C_ζ measures the distributional divergence between $P_\theta^c(\boldsymbol{x}_{inp})$ and $P_d(\boldsymbol{x})$. The entire network is trained in an end-to-end fashion. The forward path is shown using solid yellow arrows and the flow of gradients due to different terms in the objectives are also shown as color coded dashed arrows.

2. Wasserstein Loss - We use Wasserstein distance [Arjovsky et al., 2017] for D_Z (term b Eq. 4):

$$L_{Critic} = \frac{1}{s}\sum_{i=1}^{s} C_\kappa(\hat{\boldsymbol{z}}^{(i)}) - \frac{1}{s}\sum_{i=1}^{s} C_\kappa(\boldsymbol{z}^{(i)}) + \frac{\beta}{s}\sum_{i=1}^{s}\left(||\nabla^{(i)}_{\boldsymbol{z}_{avg}} C_\kappa(\boldsymbol{z}^{(i)}_{avg})||-1\right)^2 \tag{6}$$

$$L_{Gen} = -\frac{1}{s}\sum_{i=1}^{s} C_\kappa(\hat{\boldsymbol{z}}^{(i)}) \tag{7}$$

$$L_{Enc} = \frac{1}{s}\sum_{i=1}^{s} C_\kappa(\boldsymbol{z}^{(i)}) \tag{8}$$

3. Smoothing Regularization - We use Wasserstein distance [Arjovsky et al., 2017] for D_X (term c Eq. 4):

$$L_{Reg} = \frac{1}{s}\sum_{i=1}^{s} C_\zeta(\boldsymbol{x}^{(i)}_{inp}) - \frac{1}{s}\sum_{i=1}^{s} C_\zeta(\boldsymbol{x}^{(i)}) + \frac{\beta}{s}\sum_{i=1}^{s}\left(||\nabla^{(i)}_{\boldsymbol{x}_{avg}} C_\zeta(\boldsymbol{x}^{(i)}_{avg})||-1\right)^2 \tag{9}$$

$$L_{AE_{Reg}} = -\frac{1}{s}\sum_{i=1}^{s} C_\zeta(\boldsymbol{x}^{(i)}_{inp}) \tag{10}$$

Where, $\boldsymbol{z}^{(i)} = E_\phi(\boldsymbol{x}^{(i)})$, $\hat{\boldsymbol{z}}^{(i)} = G_\psi(\boldsymbol{n}^{(i)})$, $\boldsymbol{n}^{(i)} \sim \mathcal{N}(0, I)$, $z^{(i)}_{inp} = \gamma z^{(j)} + (1-\gamma) z^{(k)}$, $x^{(i)}_{inp} = D_\theta(z^{(i)}_{inp})$, $\boldsymbol{z}^{(i)}_{avg} = \alpha\boldsymbol{z}^{(i)} + (1-\alpha)\hat{\boldsymbol{z}}^{(i)}$, and $\boldsymbol{x}^{(i)}_{avg} = \alpha\boldsymbol{x}^{(i)} + (1-\alpha)\boldsymbol{x}^{(i)}_{inp}$. $\alpha \sim \mathcal{U}[0,1]$, and $\beta = 10$ is a hyper-parameter as in [Gulrajani et al., 2017]. $\gamma \sim \mathcal{U}[0,1]$. $E_\phi, D_\theta, G_\psi, C_\kappa$, and C_ζ denote the encoder, decoder, latent generator, critic, and adversarial regularization network respectively.

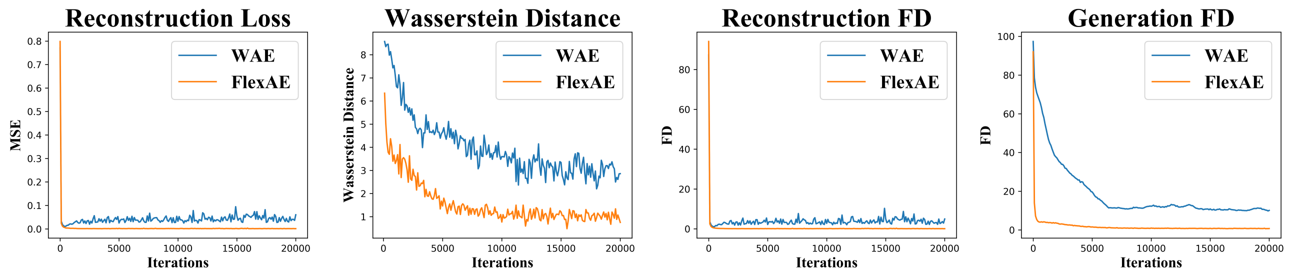
Finally, during inference, data is generated using FlexAE as follows:

1. Sample from a primitive (Gaussian) distribution and pass it through the P-GEN to sample a point from the latent space $P_\psi(\boldsymbol{z})$.
2. Input the latent sample through the Decoder to generate a data sample.

Algorithm for training FlexAE can be found in the supplementary material.

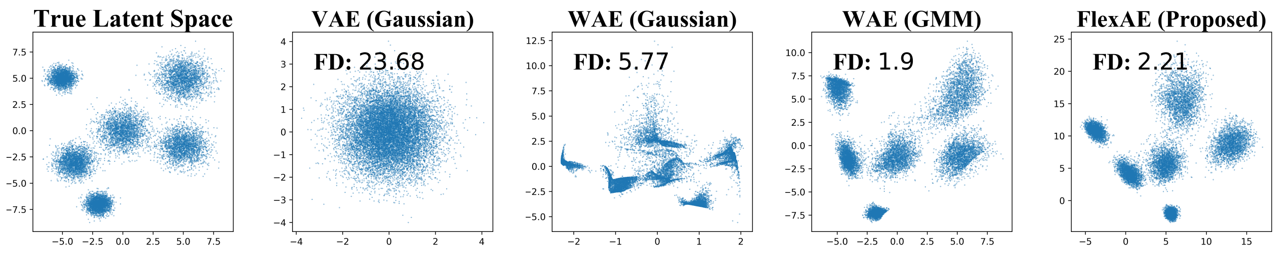
3.3 RELATION TO PRIOR WORKS

As discussed in section 2, a class of algorithms learn the prior to improve generation quality. Hoffman and Johnson [2016] proposed, learning the prior is one approach to optimize the ELBO in a stochastic VAE framework. Later, Huang et al. [2017] applied Real NVP [Dinh et al., 2017]



(a) (b) (c) (d)

Figure 2: Comparison of WAE (fixed prior) and FlexAE (learnable prior) on a synthetic dataset. Wasserstein distance between P_z and Q_ϕ reduce faster in the case of FlexAE compared to a fixed prior WAE, leading to a better FD.



(a) (b) (c) (d) (e)

Figure 3: Visualization of (a) true latent space; (b) latent space learned by the VAE [Kingma and Welling, 2014]; (c) latent space learned by the WAE [Tolstikhin et al., 2018] with Normal prior; (d) latent space learned by the WAE [Tolstikhin et al., 2018] with GMM prior; and (e): latent space learned by the proposed FlexAE model, along with generation Fréchet Distance (FD) in each case. For multimodal data, model with multimodal prior (WAE-GMM) and FlexAE perform better.

to learn the prior. Tomczak and Welling [2018] proved the optimal prior is the aggregated posterior, which they approximate by assembling a mixture of the posteriors with a set of learned pseudo-inputs. Bauer and Mnih [2019] proposed LARS prior, which is obtained by applying a rejection sampler with learned acceptance function to the original prior distribution. Takahashi et al. [2019] introduced the kernel density trick to estimate the KL divergence in ELBO and log-likelihood, without explicitly learning the aggregated posterior. However, Dai and Wipf [2019] argue that, if the data dimension, d mismatches with the model's latent dimension, m, optimizing ELBO objective leads to non-unique optimum solutions. To address this concern, they train a second stage to successfully sample from the latent space of the first stage. van den Oord et al. [2017] incorporate the ideas from vector quantisation and learns a discrete latent representation paired with a learnable autoregressive prior. On the other hand, for a deterministic encoder-decoder pair, Ghosh et al. [2020], claim that an explicit l_2 regularization in the latent space is equivalent to a Gaussian prior and leads to a smooth and meaningful latent-space. For generation, they introduce an ex-post density estimation step by fitting a ten component GMM to the learned latent space.

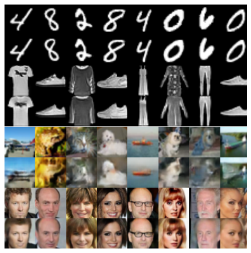
In this work, we investigate the relation between the dimensionality of the assumed latent space and fixation of a distribution as the latent prior in generative AE models with deterministic Encoder-Decoder pair, such as WAE [Tolstikhin et al., 2018], AAE [Makhzani et al., 2016]. In this perspective, we believe that ours is the first study to theoretically demonstrate the infeasibility of the objective of Generative AE under a prior fixation oblivious to true latent dimension, n.

4 EXPERIMENTS AND RESULTS

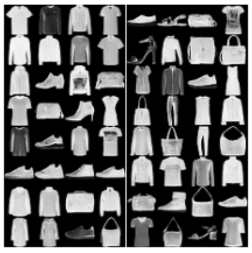
4.1 SYNTHETIC EXPERIMENTS

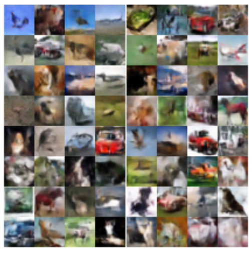
Our objective in the synthetic experiments is to study the behaviour of loss terms, reconstruction and generation quality of FlexAE and compare them with other generative AE models.

Figure 2 demonstrates the benefit of FlexAE over WAE [Tolstikhin et al., 2018], where the performance of both the models is shown on a synthetic data: $\widetilde{\mathcal{Z}} = \mathbb{R}^5$ and $f : \mathbb{R}^5 \rightarrow \mathbb{R}^{128}$ is an multi-layer perceptron with randomly initialized parameters (details in the supplementary material). It is seen that, when $m = 50$, Wasserstein distance between P_z and Q_z reduce faster and reaches much lower values in the case of FlexAE compared to a fixed prior WAE, leading to a better Fréshet Distance for generation.


(a)

(b)


(c)


(d)

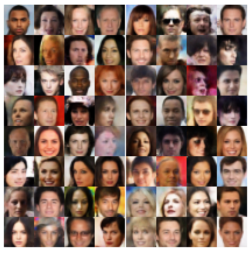

(e)

Figure 4: (a) Visualization of reconstruction quality of FlexAE-SR model on randomly selected data from the test split of MNIST (first and second rows), Fashion-MNIST (third and fourth rows), CIFAR-10 (fifth and sixth rows), and CELEBA (seventh and eights rows). The odd rows represent the real data and the even rows represent the reconstructed data. 64 Randomly generated samples (i.e. no cherry picking) of (b) MNIST, (c) Fashion MNIST, (d) CIFAR-10, and (e) CELEBA datasets using FlexAE model shows the quality of generation of the proposed model FlexAE-SR.

Table 1: Comparison of FID scores [Heusel et al., 2017] on real datasets. Lower is better.

	MNIST		Fashion		CIFAR10		CELEBA	
	Rec.	Gen.	Rec.	Gen.	Rec.	Gen.	Rec.	Gen.
VAE [Kingma and Welling, 2014]	52.14	56.07	55.90	63.01	166.45	143.31	56.66	63.85
VAE + FLOW [Kingma et al., 2016]	28.12	33.27	34.87	49.32	90.98	123.25	39.21	45.89
VAE-VampPrior [Tomczak and Welling, 2018]	21.27	55.76	31.11	50.56	109.43	141.12	51.31	60.16
VAE-IOP [Takahashi et al., 2019]	28.01	40.16	30.20	45.39	88.17	134.72	46.51	59.35
VQ-VAE [van den Oord et al., 2017]	10.95	12.07	22.47	25.73	52.11	85.98	34.23	42.15
WAE-GAN [Tolstikhin et al., 2018]	10.48	13.60	25.93	30.21	51.36	92.30	32.49	45.58
2-S VAE [Dai and Wipf, 2019]	10.83	11.91	22.43	28.57	73.07	94.57	37.60	44.85
AE + GMM (L2) [Ghosh et al., 2020]	9.69	14.70	21.59	26.74	51.45	95.77	30.16	43.79
RAE + GMM (L2) [Ghosh et al., 2020]	9.25	10.80	19.71	25.50	50.84	90.40	34.35	44.72
MaskAAE [Mondal et al., 2020]	9.53	10.12	21.59	27.29	71.40	88.82	39.75	46.79
Pioneer [Heljakka et al., 2018]	8.17	7.30	15.91	17.25	51.07	59.12	25.35	27.94
ALAE [Pidhorskyi et al., 2020]	7.97	7.07	13.25	16.87	50.93	57.36	24.39	26.15
FlexAE (Proposed)	8.28	7.28	14.68	14.81	69.63	74.45	35.31	36.99
FlexAE-SR (Proposed)	**7.63**	**6.75**	**14.17**	**14.41**	**50.36**	**54.61**	**23.94**	**25.78**
Best GAN from Lucic et al. [2018]	-	~ 6	-	~ 20	-	~ 55	-	~ 30

Further, to demonstrate the effect of having a learnable prior, we plot the latent spaces learned using several AE models in Figure 3 on a synthetic dataset ($\widetilde{\mathcal{Z}} = \mathbb{R}^2$ and $f : \mathbb{R}^2 \to \mathbb{R}^{128}$, further details in the supplementary material) where the true latent space is a mixture of Gaussian (MoG) . It can be seen that the latent space learned by a FlexAE and a WAE with a GMM prior results in better generation as compared to the models with fixed uni-modal Gaussian priors (Note that this figure is to show that a flexible prior helps in learning but not to show impossibility).

4.2 REAL-WORLD DATASETS

We consider four real-world datasets: MNIST [Lecun, 2010], Fashion-MNIST [Xiao et al., 2017], CIFAR-10 [Krizhevsky, 2009], and CelebA [Liu et al., 2015] for our three sets of experiments. We have adopted the architecture as in 2S-VAE [Dai and Wipf, 2019] (refer to the supp.) for the AE across all the experiments for the proposed method and all baseline models to ensure a fair comparison.

4.2.1 Baseline Experiments

Methodology: The first task is to evaluate the FlexAE as a generative model. We use Fréchet Inception Distance, (FID) [Heusel et al., 2017], one of the most commonly used evaluation methods as it correlates well with human visual perception [Lucic et al., 2018]. However, as observed in [Sajjadi et al., 2018], FID, being uni-dimensional, fails to distinguish between different cases of failure (poor sample quality and limited variation in the samples). Thus, we also report the precision and recall metrics described in [Sajjadi et al., 2018] along with FID, both of which are computed between the generated and the real test images. We compare FlexAE with a number of SoTA AE-based generative models that cover a broad class namely, VAE [Kingma and Welling, 2014], VAE+Flow [Kingma et al., 2016], VAE-VampPrior [Tomczak and Welling, 2018], VAE-IOP [Takahashi et al., 2019], VQ-VAE [van den Oord et al., 2017], WAE [Tolstikhin et al., 2018], a plain with AE post-hoc GMM, RAE+GMM [Ghosh et al., 2020], 2-stage VAE [Dai and Wipf, 2019], MaskAAE [Mondal et al., 2020], Pi-

Table 2: Comparison of Precision/Recall scores [Sajjadi et al., 2018] on real datasets. Higher is better.

	MNIST	Fashion	CIFAR10	CELEBA
VAE [Kingma and Welling, 2014]	0.68/0.74	0.61/0.66	0.28/0.44	0.48/0.57
2S-VAE [Dai and Wipf, 2019]	0.95/0.98	0.80/0.78	0.44/0.61	0.72/0.78
RAE + GMM (L2) [Ghosh et al., 2020]	0.96/0.97	0.78/0.77	0.49/0.59	0.67/0.75
MaskAAE [Mondal et al., 2020]	0.94/0.96	0.71/0.83	0.45/0.62	0.65/0.62
ALAE [Pidhorskyi et al., 2020]	**0.98/0.99**	0.97/0.96	0.73/0.85	0.84/0.81
FlexAE (Proposed)	**0.98/0.99**	**0.98**/0.98	0.65/0.80	0.71/0.76
FlexAE-SR (Proposed)	**0.98/0.99**	**0.98/0.99**	**0.79/0.87**	**0.85/0.88**

oneer [Heljakka et al., 2018], and ALAE [Pidhorskyi et al., 2020] with same (or similar capacity) architectures (see supplementary material). Note that a single data agnostic setting for all the hyper-parameters have been chosen for all the experiments related to FlexAE and FlexAE-SR (Refer to supplementary material for details). The details of the P-GEN and Smoothing Regularization networks can also be found in the supplementary.

Results: Table 1 compares the average reconstruction and generation FID scores (lower is better) of FlexAE with other AE-based generative models and the best GAN's generation FID as reported in [Lucic et al., 2018]. It is seen that while models with parametric learnable priors (VampPrior, IOP, Flow) offer some improvement over the naive VAE, they are non optimum. It is also seen that complex prior models tend to over fit more (gap between the gen. and recon. FIDs). Further, controlling the latent space dimensionality (2SVAE, MaskAAE) have significant impact on the performance. A relatively better performance of RAE+GMM shows that while absence of prior imposition will reduce the bias, it might lead to over fitting. Pioneer [Heljakka et al., 2018] with progressively growing training procedure and adversarial encoder-generator network shows significant boost in performance[2]. ALAE [Pidhorskyi et al., 2020] with their modified generator and discriminator architecture offers slightly better performance as compared to Pioneer. Finally, FlexAE-SR offers the best performance on three datasets as compared to other AE based generative models and their performance are comparable to that of the GANs [Lucic et al., 2018]. A similar trend is observed with the Precision/Recall scores in Table 2. FlexAE-SR achieves significantly better numbers confirming its effectiveness in generating samples that are of both high quality and variety.

Figure 4 presents reconstructed and generated samples using FlexAE-SR for qualitative evaluation of its performance. Supplementary material contains more examples.

[2]To ensure fairness in comparison, in this work all the methods were trained for same number of iterations.

4.2.2 Effect of Latent Space Dimensionality

Methodology: To study the effect of the latent dimensionality on the generation quality of the latent variable models, we train FlexAE-SR and WAE models with varying m.

Results: As presented in Table 3, with increasing m, the reconstruction FID decreases for both WAE and FlexAE. However, the generation FID of WAE increases with m. While generation FID of FlexAE remains almost constant. This shows that FlexAE can achieve better optimum irrespective of the latent dimensionality.

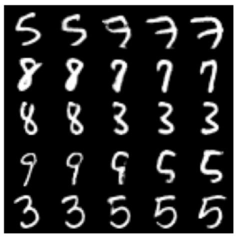
4.2.3 Smoothness of the Latent Space

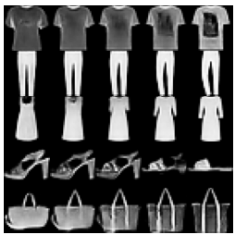
Qualitative Comparison: To ascertain the smoothness of the learned latent space and that FlexAE-SR doesn't over fit, we conduct a few qualitative experiments: (i) traversal in the latent space between two real samples (MNIST and FMNIST) (ii) Generation by transitions in the latent space along the direction of a particular attribute (CELEBA) (iii) plot of the Nearest neighbour samples for a given generated image, from the training set, based on pixel values and inception features (CELEBA).

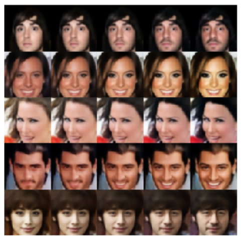
The outcome of these experiments are shown in Figure 5. Each row in 5a and 5b represents linear interpolation in the latent space between two randomly selected samples in the first and the last column. The central image in each row is generated using the mid-point of the two latent representations corresponding to the two real images. Each row in 5c presents manipulation of a particular face attribute (Big Nose, Heavy Makeup, Black Hair, Smiling, Male). The middle image in each row of 5c corresponds to a training sample with the attribute present. The interpolation results presented in 5a, 5b, and 5c clearly depict the smoothness of FlexAE-SR latent space as it provides smooth transition between any two random images (5a, 5b) or smooth transition based on a feature (5c). The first image of each row in 5d, and 5e shows a randomly generated sample using FlexAE-SR and the next four entries are the four nearest neighbours from the training split based on raw pixel values and inception features respectively. Visual dissimilarity between a generated image and its nearest neighbours in

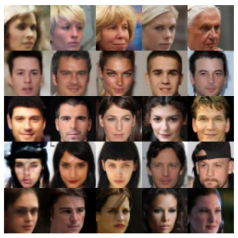
Table 3: Variation of FID w.r.t. bottleneck layer dimension, m. For MNIST and Fashion MNIST $m_b = 32$ and for CIFAR-10 and CELEBA $m_b = 64$. Unlike WAE [Tolstikhin et al., 2018], generation quality of FlexAE-SR remains unaltered even when m increases.

m	MNIST				FASHION				CIFAR10				CELEBA			
	Rec.		Gen.		Rec.		Gen.		Rec.		Gen.		Rec.		Gen.	
	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR	WAE	FlexAE-SR
m_b	7.98	6.19	15.34	5.86	20.71	10.86	40.5	11.46	51.36	50.36	92.3	54.61	32.49	23.94	45.58	25.78
$2m_b$	5.61	3.10	24.58	3.37	13.94	7.20	55.49	8.41	48.9	43.59	87.91	53.94	25.72	18.58	87.1	21.26
$4m_b$	3.99	1.64	37.16	2.79	10.04	4.93	78.31	8.16	30.47	29.71	89.82	56.09	19.89	16.21	76.12	22.18


(a)


(b)


(c)


(d)

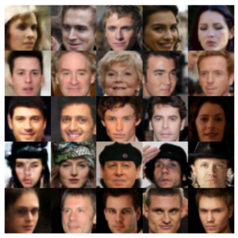

(e)

Figure 5: (a), (b) Each row in represents linear interpolation in the latent space between two randomly selected test samples in the first and the last entry. (c) Each row presents manipulation of a particular face attribute (Big Nose, Heavy Makeup, Black Hair, Smiling, Male). The central image of each row is a true image from the train split with the attribute. (d) The first image in each row shows randomly generated samples using FlexAE-SR and the next four entries are the four nearest neighbours from training data. (e) First entry in each row shows the same randomly generated samples as in (d) and the four nearest neighbours based on features extracted by a pretrained Inception net.

the training set confirms that FlexAE-SR has not merely memorized the training set. (cf. supp. for more results).

Quantitative Comparison: To quantitatively understand, if the proposed method is merely memorizing the training examples, we propose two metrics: 1) Pixel Memorization Score (PMS) and 2) Inception Memorization Score (IMS). Given two sets of images A and B, PMS computes the average L2 distance in the pixel space of all 1-nearest neighbours of images in the set A from those in the set B. IMS is a similar distance metric in the inception feature space. Table 4 shows the results. We can see that, PMS between two non-overlapping sets of $10k$ training(Tr) / test(Te) images are comparable to PMS between $10k$ generated samples and $10k$ training samples. It implies that the generated samples are not replicas of the training samples seen by the model. Otherwise, PMS(Gen, Tr) would be less than PMS(Tr, Tr). Similar argument holds true for the proposed IMS metric. This shows the model has not memorized the training samples.

Table 4: Pixel/Inception Memorization Score

	Tr-Tr	Gen-Tr	Tr-Te	Gen-Te
MNIST	4.72/7.11	4.66/7.10	4.77/7.12	4.66/7.12
Fashion	4.01/9.22	3.99/9.13	4.01/9.26	3.90/9.16
CIFAR-10	9.75/12.54	9.64/14.40	9.74/12.51	9.68/14.40
CELEBA	18.42/8.73	18.01/9.37	18.50/8.75	17.86/9.34

5 CONCLUSION

In this paper, we systematically studied the effect of the latent prior on the deterministic AE-based generative models. We demonstrated that fixing any kind of prior in a data-agnostic way is detrimental to the performance. We proposed a model called the FlexAE, where we have introduced an additional state space to address the problem of infeasibility that arises due to latent dimensionality mismatch and prior fixation. We also employ a smoothing regularization technique to learn a locally convex smooth latent space for deterministic generative autoencoders. We have empirically demonstrated the efficacy of the proposed models on several real world datasets.

Author Contributions

Arnab Kumar Mondal led the work and performed all the experiments. The other authors contributed in refining the core ideas and helped in developing the theory. All the authors contributed in writing the paper.

Acknowledgements

We thank IIT Delhi HPC facility[3] for computational resources. Parag Singla is supported by the DARPA Explainable Artificial Intelligence (XAI) Program with number

[3]http://supercomputing.iitd.ac.in

N66001-17-2-4032, the Visvesvaraya Young Faculty Fellowships by Govt. of India and IBM SUR awards. Himanshu Asnani acknowledges the support of Department of Atomic Energy, Government of India, under the project no. 12-R&D-TFR-5.01-0500 and a gift from Adobe Research. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of the funding agencies.

References

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In Proc. of ICLR, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Proc. of ICML, 2017.

Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. In Proc. of AISTATS, 2019.

David Berthelot, Colin Raffel, Aurko Roy, and Ian Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. In Proc. of ICLR, 2019.

Olivier Bousquet, Sylvain Gelly, Ilya Tolstikhin, Carl-Johann Simon-Gabriel, and Bernhard Schoelkopf. From optimal transport to generative modeling: the vegan cookbook. arXiv preprint arXiv:1705.07642, 2017.

Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β -VAE. In NeuRIPS Workshop, 2017.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Proc. of NeurIPS, 2016.

Bin Dai and David Wipf. Diagnosing and enhancing vae models. In Proc. of ICLR, 2019.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In Proc. of ICLR, 2017.

Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From variational to deterministic autoencoders. In Proc. of ICLR, 2020.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In Proc. of NeuRIPS, 2017.

Ari Heljakka, Arno Solin, and Juho Kannala. Pioneer networks: Progressively growing generative autoencoder. In Proc. of ACCV, 2018.

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 Proc. of NeuRIPS, 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In Proc. of ICLR, 2017.

Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In NeurIPS Workshop, 2016.

Chin-Wei Huang, Ahmed Touati, Laurent Dinh, Michal Drozdzal, Mohammad Havaei, Laurent Charlin, and Aaron Courville. Learnable explicit density for continuous latent space and variational inference. arXiv preprint arXiv:1710.02248, 2017.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Proc. of ICML, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In Proc. of ICLR, 2014.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In Proc. of NeuRIPS, 2016.

Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in VAEs. In Proc. of NeuRIPS, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Abhishek Kumar, Ben Poole, and Kevin Murphy. Regularized autoencoders via relaxed injective probability flow. In Proc. of AISTATS, 2020.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In Proc. of ICML, 2016.

Y. Lecun. The mnist database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`, 2010.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proc. of ICCV, 2015.

Mario Lucic, Karol Kurach, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Are gans created equal? a large-scale study. In Proc. of NeuRIPS, 2018.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. In Proc. of ICLR, 2016.

Arnab Kumar Mondal, Sankalan Pal Chowdhury, Aravind Jayendran, Parag Singla, Himanshu Asnani, and AP Prathosh. MaskAAE: Latent space optimization for adversarial auto-encoders. In Proc. of UAI, 2020.

Eric Nalisnick and Padhraic Smyth. Stick-breaking variational autoencoders. In Proc. of ICLR, 2017.

Eric Nalisnick, Lars Hertel, and Padhraic Smyth. Approximate inference for deep latent gaussian mixtures. In NeurIPS Workshop, 2016.

Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In Proc. of CVPR, 2020.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Proc. of ICML, 2015.

Danilo Jimenez Rezende and Fabio Viola. Taming vaes. arXiv preprint arXiv:1810.00597, 2018.

Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. Distribution matching in variational inference. arXiv preprint arXiv:1802.06847, 2018.

Paul K. Rubenstein, Bernhard Schoelkopf, and Ilya Tolstikhin. Wasserstein auto-encoders: Latent dimensionality and random encoders. In Proc. of ICLR Workshop, 2018.

Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In Proc. of NeuRIPS, 2018.

Jiaming Song Shengjia Zhao and Stefano Ermon. Towards deeper understanding of variational autoencoding models. arXiv preprint arXiv:1702.08658, 2017.

Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, and Satoshi Yagi. Variational autoencoder with implicit optimal priors. In Proc. of AAAI, 2019.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Scholkopf. Wasserstein auto-encoders. In Proc. of ICLR, 2018.

Jakub M Tomczak and Max Welling. VAE with a vampprior. In Proc. of AISTATS, 2018.

Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In Proc. of NeuRIPS, pages 6306–6315, 2017.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In Proc. of ICML, 2019.

Davis Wertheimer, Omid Poursaeed, and Bharath Hariharan. Augmentation-interpolative autoencoders for unsupervised few-shot image generation. arXiv preprint arXiv:2011.13026, 2020.

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

Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Balancing learning and inference in variational autoencoders. In Proc. of AAAI, 2019.

6 THEORY

WAE Objective:

$$D_{WAE} = \inf_{\phi,\theta}\left(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})}\mathop{\mathbb{E}}_{Q_\phi(\boldsymbol{z}|\boldsymbol{x})}\Big[c\Big(\boldsymbol{x}, D_\theta\big(E_\phi(\boldsymbol{x})\big)\Big)\Big]}_{\text{a}} + \lambda \cdot \underbrace{D_Z\big(Q_\phi(\boldsymbol{z}), P_Z(\boldsymbol{z})\big)}_{\text{b}}\right) \tag{11}$$

FlexAE Objective:

$$D_{FlexAE} = \inf_{\psi,\phi,\theta}\left(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})}\mathop{\mathbb{E}}_{Q(\boldsymbol{z}|\boldsymbol{x})}\Big[c\big(\boldsymbol{x}, D_\theta(\boldsymbol{z})\big)\Big]}_{\text{a}} + \lambda \cdot \underbrace{D_Z\big(Q_\phi(\boldsymbol{z})||P_\psi(\boldsymbol{z})\big)}_{\text{b}}\right) \tag{12}$$

FlexAE-SR Objective:

$$D_{FlexAE-SR} = \inf_{\psi,\phi,\theta}\left(\underbrace{\mathop{\mathbb{E}}_{P_d(\boldsymbol{x})}\mathop{\mathbb{E}}_{Q(\boldsymbol{z}|\boldsymbol{x})}\Big[c\big(\boldsymbol{x}, D_\theta(\boldsymbol{z})\big)\Big]}_{\text{a}} + \lambda_1 \cdot \underbrace{D_Z\big(Q_\phi(\boldsymbol{z})||P_\psi(\boldsymbol{z})\big)}_{\text{b}} + \lambda_2 \cdot \underbrace{D_X\big(P_\theta(\boldsymbol{x}_{inp})||P_d(\boldsymbol{x})\big)}_{\text{c}}\right) \tag{13}$$

Theorem 2. *If $m > n$, then the regularization term in the objective function of a WAE (Eq. 11), $D_Z(Q_\phi(\boldsymbol{z}), P_Z(\boldsymbol{z})) > 0$, $\forall\phi$ and for any distributional divergence D_Z when the support of $P_Z(\boldsymbol{z}) \notin \mathcal{Q}^n_m$.*

Proof. Since $f : \widetilde{\mathcal{Z}} \to \mathcal{X}$ can be approximated arbitrarily closely using a neural network (the assumption we have made earlier) and the Encoder function $E_\phi : \mathcal{X} \to \mathcal{Z}$ is also a neural network, $E_\phi \circ f : \mathbb{R}^n \to \mathbb{R}^m$ belongs to the class of composition of affine transformations and point wise non-linearities (such as rectifiers, leaky rectifiers, or smooth strictly increasing functions like sigmoid, tanh, softplus, etc.). Consequently, $\mathcal{Z}$ is always a countable union of n-dimensional manifolds in a m-dimensional ambient space (Lemma 1 in Arjovsky and Bottou [2017]). Therefore, given that the Encoder is deterministic, by definition, $Q_\phi(\boldsymbol{z})$ has measure zero on $\mathbb{R}^m \backslash \mathcal{Z}$, whereas the support of $P_Z(\boldsymbol{z}) \notin \mathcal{Q}^n_m$ which implies that it has a non-zero measure outside $\mathcal{Z}$. Thus, any distributional divergence measure between $Q_\phi(\boldsymbol{z})$ and $P_Z(\boldsymbol{z})$ will assume a non-zero value, whenever $m > n$. □

Corollary 2.1. *$\forall m' \geq n$, $D_Z(Q_\phi(\boldsymbol{z})||P_\psi(\boldsymbol{z}))$ (term (b) in FlexAE objective (Eq. 12) becomes zero for optimum set of parameters.*

Proof. Let, $\mathcal{P}_\psi$ denote the set of all possible manifolds on which the output of P-GEN network, G_ψ, may lie within $\mathbb{R}^m$. Given sufficiently large deep nets, sample size, and computation time, $\mathcal{P}_\psi = \cup_{\eta \leq m'} \mathcal{Q}^\eta_m$. As $m' \geq n$, this implies $\mathcal{Q}^n_m \subseteq \mathcal{P}_\psi$ which implies that G_ψ can learn $P_\psi(\boldsymbol{z})$ to match $Q_\phi(\boldsymbol{z})$ driving $D_Z(Q_\phi(\boldsymbol{z})||P_\psi(\boldsymbol{z}))$ to zero. □

7 DETAILS OF DATASETS

In this section, we describe the steps involved in synthetic dataset creation and provide relevant details (such as dimension, number of training and test examples and so on) of the synthetic datasets and the real datasets used in our work to experimentally validate our theoretical claims.

7.1 SYNTHETIC DATASETS

Synthetic data has been generated using a two step process. The steps involved in creating the dataset (corresponding to Figure 2 in the main paper) where the true latent space is a Mixture of Gaussian (MoG) are listed below.

1. **Step 1:** Six five-dimensional Gaussian distributions are used to generate true latent space of the synthetic dataset. $z^{(i)}_{k1}$, $z^{(i)}_{k2}$, $z^{(i)}_{k3}$, $z^{(i)}_{k4}$, and $z^{(i)}_{k5}$ denotes the 1^{st}, 2^{nd}, 3^{rd}, 4^{th} and 5^{th} dimensions of the i^{th} sample from the k^{th} component. The distributions are as mentioned below:

$$\begin{bmatrix} z^{(i)}_{11} \\ z^{(i)}_{12} \\ z^{(i)}_{13} \\ z^{(i)}_{14} \\ z^{(i)}_{15} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z^{(i)}_{21} \\ z^{(i)}_{22} \\ z^{(i)}_{23} \\ z^{(i)}_{24} \\ z^{(i)}_{25} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 5 \\ 5 \\ 5 \\ 5 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z^{(i)}_{31} \\ z^{(i)}_{32} \\ z^{(i)}_{33} \\ z^{(i)}_{34} \\ z^{(i)}_{35} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -5 \\ 5 \\ 3 \\ 4.5 \\ -6 \end{bmatrix}, 0.5 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{41}^{(i)} \\ z_{42}^{(i)} \\ z_{43}^{(i)} \\ z_{44}^{(i)} \\ z_{45}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ -1.5 \\ -6.5 \\ 3 \\ 1 \end{bmatrix}, 0.95 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{51}^{(i)} \\ z_{52}^{(i)} \\ z_{53}^{(i)} \\ z_{54}^{(i)} \\ z_{55}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -2 \\ -7 \\ 9 \\ -1.5 \\ -4.5 \end{bmatrix}, 0.5 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{61}^{(i)} \\ z_{62}^{(i)} \\ z_{63}^{(i)} \\ z_{64}^{(i)} \\ z_{65}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -4 \\ -3 \\ -5.5 \\ 2 \\ 4 \end{bmatrix}, 0.75 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right)$$

2. **Step 2:** Next, a three layer MLP is used to map the five-dimensional points obtained from Step 1 to 128-dimensional data points. Each layer consists of 128 neurons and non-linearity used in each layer is $\tanh, \exp, \tanh$ respectively. Weight and bias parameters of each layer is drawn randomly from the following three distributions respectively: $\mathcal{N}(0, 0.05), \mathcal{N}(0, 0.2), \mathcal{N}(0, 0.1)$.

The synthetic dataset related to Figure 3 in the main paper is also generated using a 6 component GMM latent space. However, the true latent has dimension 2 and synthetic data has dimension 128 as before. The mean and variance of the Gaussian components of the true latent space are listed below.

$$\begin{bmatrix} z_{11}^{(i)} \\ z_{12}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{21}^{(i)} \\ z_{22}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 5 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{31}^{(i)} \\ z_{32}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -5 \\ 5 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{41}^{(i)} \\ z_{42}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ -1.5 \end{bmatrix}, \begin{bmatrix} 0.95 & 0 \\ 0 & 0.95 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{51}^{(i)} \\ z_{52}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -2 \\ -7 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\right)$$

$$\begin{bmatrix} z_{61}^{(i)} \\ z_{62}^{(i)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -4 \\ -3 \end{bmatrix}, \begin{bmatrix} 0.75 & 0 \\ 0 & 0.75 \end{bmatrix}\right)$$

We have generated 15k training examples and 10k test examples for both of the synthetic datasets.

7.2 REAL DATASETS

The MNIST Lecun [2010] database of gray scale handwritten digits consists of 60000 training examples and 10000 test samples. Fashion-MNIST Xiao et al. [2017] is a drop-in replacement for the original MNIST dataset. It consists of 60000 and 10000 gray-scale images in the training and test splits respectively from Zalando's article images. The CIFAR-10 Krizhevsky [2009] dataset consists of 60000 tiny RGB images from 10 classes, with 6000 images per class. The standard split of this dataset consists of 50000 training images and 10000 test images. For experiments with MNIST and CIFAR-10, we use datasets as provided by Tensorflow API. CelebFaces Attributes Dataset (CelebA) Liu et al. [2015] is a large-scale face attributes dataset with 202599 celebrity images, each with 40 attribute annotations. For experiments with CELEBA, we resize the images to 64×64 following many prior works Kumar et al. [2020], Mondal et al. [2020], Dai and Wipf [2019], Ghosh et al. [2020] in generative model. Table 5 summarizes the important information about the real datasets used in this paper. Although, the test split of CELEBA dataset contains more than 10k examples, we use 10k randomly selected samples for FID and precision/recall score computation for all the datasets.

Table 5: Details of Real Datasets

	Dimension ($h \times w \times c$)	Train Split Size	Test Split Size
MNIST Lecun [2010]	$28 \times 28 \times 1$	60000	10000
Fashion-MNIST Xiao et al. [2017]	$28 \times 28 \times 1$	60000	10000
CIFAR-10 Krizhevsky [2009]	$32 \times 32 \times 3$	50000	10000
CELEBA Liu et al. [2015]	$64 \times 64 \times 3$	162770	19962

8 NETWORK ARCHITECTURES

Like any other AE based generative model, FlexAE has a reconstruction pipeline consisting of an encoder (E_ϕ) and a decoder (D_θ) network. We have introduced a P-GEN network consisting of a generator network (G_ψ) and a critic network (C_κ) to facilitate sampling from the latent space of the reconstruction pipeline. The generation pipeline involves the latent generator, G_ψ and the image generator, D_θ, meaning generation is a two-step process. First, we sample from the latent space using the latent generator, G_ψ. Next, the image generator, D_θ samples from the image space using the generated latent code. FlexAE-SR consists of one additional critic network, C_ζ, which acts as a smoothing regularization network.

Next, we describe the architectures of each of the components of FlexAE used for the synthetic and the real experiments.

8.1 ARCHITECTURES FOR SYNTHETIC EXPERIMENTS

Table 6 presents architectures of different networks used in conducting the synthetic experiment. VAE Kingma and Welling [2014] consists of only encoder and decoder. WAE Tolstikhin et al. [2018] consists of encoder, decoder and critic. FlexAE involves all the networks.

8.2 ARCHITECTURES FOR EXPERIMENTS ON REAL-WORLD DATASETS

For real experiments, the encoder, E_ϕ and the decoder, D_θ architectures are adopted from prior works Chen et al. [2016], Lucic et al. [2018], Dai and Wipf [2019]. The architecture of the encoder and the decoder networks are same irrespective of the dataset chosen, as presented in Table 7. However, number of neurons in the 2^{nd} fully connected layer of decoder varies based on the dimension of the images. The architectures of the generator, G_ψ, the critic, C_κ and the smoothing regularization network, C_ζ are fixed across all datasets as mentioned in Table 8.

Notation wise, CONV$_{n,k,s}$ denotes a convolutional layer with n kernels of size k and stride size s. TCONV$_{n,k,s}$ denotes a transpose convolutional layer with n kernels of size k and stride size s. FC$_n$ denotes a Fully Connected layer with n neurons. BN denotes a batch normalization layer. $ReLU$ denotes Rectified Linear Unit activation.

To study the effect to latent space dimensionality, m on the generation quality, we train different FlexAE-SR models with varying m while everything else is kept fixed.

9 TRAINING ALGORITHM, HYPER-PARAMETERS, COMPUTING RESOURCES AND AVERAGE RUNTIME

As mentioned in the main paper, the auto-encoder is required to be optimized jointly with the P-GEN to ensure regularization in the AE latent space. This regularization effectively enforces smoothness in the learnt latent space and prevents the AE from overfitting on the training examples. In order to be able to satisfy the above requirement in practice, we optimize each of the four losses specified in the main paper in every training iteration. Specifically, in each learning loop, we optimize the L_{AE}, L_{Critic}, L_{Reg}, $L_{AE_{Reg}}$, L_{Gen}, and L_{Enc} in that order using a learning schedule. We use Adam optimizer for our optimization. The training algorithm is described in Algorithm 1. For real experiments we have trained our models for 130000 iterations on each dataset with a batch size of 128. We have used a machine with Intel® Xeon® Gold 6142 CPU, 376GiB RAM, and Zotac GeForce® GTX 1080 Ti 11GB Graphic Card for all of our experiments. The average runtime for experiments on MNIST, Fashion-MNIST, CIFAR-10, and CELEBA is approximately 15 hours, 15 hours, 20 hours, and 44 hours respectively.

10 EXPERIMENTAL RESULTS

In the main paper, the performance of FlexAE is evaluated mainly quantitatively, using standard metrics: FID Heusel et al. [2017] and precision/recall Sajjadi et al. [2018] score. We have used 10000 reconstructed and 10000 generated samples against 10000 test examples for computation of FID and precision/recall score for all datasets. It has been observed that both FlexAE and FlexAE-SR outperform all other current state-of-the-art AE based generative models as measured using those metrics. In this section, we present more qualitative results (generated samples and decoded images due to interpolation in the latent space) for visual evaluation of the proposed generative framework, FlexAE-SR.

Figure 6, 7 8, 9 presents 225 randomly generated samples of MNIST, Fashion-MNIST, CIFAR-10 and CELEBA datasets respectively.

Next, we present images obtained by decoding latent vectors obtained by linear combinations of the encoded latent vectors of two real images from a given dataset. The first and the last entries of each row in Figure 10 are the reconstruction of two real images from the respective datasets.

Table 6: Network Architectures for Synthetic Experiment

Encoder	Decoder	Generator	Critic
$\boldsymbol{x} \in \mathbb{R}^{128}$	$\boldsymbol{z} \in \mathbb{R}^{2}$	$\boldsymbol{n} \in \mathbb{R}^{2}$	$\boldsymbol{z} \in \mathbb{R}^{2}$
$\to$ FC$_{128}$ $\to$ ReLU	$\to$ FC$_{128}$ $\to$ Tanh	$\to$ FC$_{128}$ $\to$ ReLU	$\to$ FC$_{128}$ $\to$ ReLU
$\to$ FC$_m$		$\to$ FC$_m$	$\to$ FC$_{128}$ $\to$ ReLU
			$\to$ FC$_1$
$m = 50$ for the first synthetic experiment (Figure 2 in the main paper) and $m = 2$ for the second synthetic experiment (Figure 3 in the main paper).			

Table 7: Encoder and Decoder Architectures for Real Datasets

Encoder, (E_ϕ)	Decoder, (D_θ)
$\boldsymbol{x} \in \mathbb{R}^{h\times w\times c}$	$\boldsymbol{z} \in \mathbb{R}^{m}$
$\to$ Conv$_{64,4,2}$ $\to$ ReLU	$\to$ FC$_{1024}$ $\to$ BN $\to$ ReLU
$\to$ Conv$_{128,4,2}$ $\to$ BN $\to$ ReLU	$\to$ FC$_{\frac{h}{4}\times\frac{w}{4}\times 128}$ $\to$ BN $\to$ ReLU
$\to$ Flatten $\to$ FC$_{1024}$ $\to$ BN $\to$ ReLU	$\to$ Reshape$_{\frac{h}{4}\times\frac{w}{4}\times 128}$
$\to$ FC$_m$	$\to$ TCONV$_{128,4,2}$ $\to$ BN $\to$ ReLU
	$\to$ TCONV$_{64,4,2}$ $\to$ BN $\to$ ELU
	$\to$ CONV$_{c,4,1}$ $\to$ Sigmoid
$m = 20$ for MNIST, Fashion-MNIST and $m = 64$ for CIFAR10, CELEBA.	

Table 8: Generator, Critic and Smoothing Regularization Network Architectures for Real Datasets

Generator, (G_ψ)	Critic, (C_κ)	Smoothing Regularization Network, C_ζ
$\boldsymbol{n} \in \mathbb{R}^{m}$	$\boldsymbol{z} \in \mathbb{R}^{m}$	$\boldsymbol{x} \in \mathbb{R}^{h\times w\times c}$
$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ Conv$_{64,4,2}$ $\to$ ReLU
$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ Conv$_{128,4,2}$ $\to$ ReLU
$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ Flatten $\to$ FC$_{1024}$ $\to$ ReLU
$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ FC$_{1024}$ $\to$ ReLU	$\to$ FC$_1$
$\to$ FC$_m$	$\to$ FC$_1$	
$m = 20$ for MNIST, Fashion-MNIST and $m = 64$ for CIFAR-10, CELEBA.		

Algorithm 1 Pseudo code for the training loop of FlexAE-SR

Hyper-parameters: $\eta_{AE} = 0.0002$, $\eta_{Critic} = 0.0001$, $\eta_{Reg} = 0.0002$, $\eta_{AE_{Reg}} = 0.0002$, $\eta_{Gen} = 0.0001$, $\eta_{Enc} = 0.00001$, AE_OPT = Adam(lr = $\eta_{AE}, \beta_1 = 0.9, \beta_2 = 0.999$), CRITIC_OPT = Adam(lr = $\eta_{Critic}, \beta_1 = 0.0, \beta_2 = 0.9$), REG_OPT = Adam(lr = $\eta_{Reg}, \beta_1 = 0.5, \beta_2 = 0.9$), AE_REG_OPT = Adam(lr = $\eta_{AE_{Reg}}, \beta_1 = 0.5, \beta_2 = 0.9$), GEN_OPT = Adam(lr = $\eta_{Gen}, \beta_1 = 0.0, \beta_2 = 0.9$), ENC_OPT = Adam(lr = $\eta_{Enc}, \beta_1 = 0.0, \beta_2 = 0.9$), $disc\_training\_ratio = 5$.

```
1:  function TRAIN
2:      for $i \leftarrow 1$ to $training\_steps$ do
3:          Minimize $L_{AE}$ and Update $\phi,\ \theta$
4:          for $j \leftarrow 1$ to $disc\_training\_ratio$ do
5:              Minimize $L_{Critic}$ and Update $\kappa$
6:              Minimize $L_{Reg}$ and Update $\zeta$
7:          end for
8:          Minimize $L_{AE_{Reg}}$ and Update $\phi,\ \theta$
9:          Minimize $L_{Gen}$ and Update $\psi$
10:         Minimize $L_{Enc}$ and Update $\phi$
11:     end for
12: end function
```

The remaining images in between the first and the last images are decoded from interpolated latent vectors. The middle image of each row is decoded from the average latent code of the encodings of the two real images. Figure 10a presents linear interpolation between two different digits from MNIST dataset. Figure 10b shows linear interpolation between two fashion items (Fashion-MNIST) either from the same class or from two different classes. Figure 10c captures linear interpolation between two different objects of the same class from the CIFAR-10 dataset. Figure 10d shows linear interpolation between two randomly selected face from CELEBA dataset. Irrespective of the coefficient of convex combination, the decoded images obtained from the interpolated latent codes look realistic in most of the cases. This indicates that the learned latent space of the FlexAE-SR model is smooth.

Next, we present more attribute based interpolation results from the CELEBA test split in Figure 11, Figure 12, Figure 13, Figure 14, and Figure 15 for the attributes “Big Nose”, “Heavy Makeup”, “Black Hair”, “Smiling”, and “Male” respectively. The central image of the grid in the sub-figures (a) and (b) in every figure presents the reconstruction of a negative example from the CELEBA dataset i.e. a sample without the corresponding attribute. Whereas, the central image in the grid of the sub-figures (c) and (d) presents the reconstruction of a positive example i.e. a sample with the particular attribute. For latent space traversal along a particular attribute direction, we calculate the average representation, z_{pos} with respect to all the positive training samples and the average representation, z_{neg} with respect to all the negative training samples. Finally, we use the direction $(z_{pos} - z_{neg})$ to traverse the latent space for attribute manipulation. Please note, this supervised traversal is performed post training in order to understand if the trained model could learn the meaning of different face attributes without supervision. The training was completely unsupervised without using any label information. As can be seen from the Figures 11 - 15, FlexAE could successfully learn the concept of different attributes without any kind of supervision. Otherwise, the interpolated figures would not be so smooth.

Finally, Figure 16 presents a 15×15 grid, where, the first column plots some randomly generated CIFAR-10 images and the remaining entries in each row are the 14 nearest neighbours (in terms of Euclidean distance) based on pixel values from the training split. Figure 17 presents another 15×15 grid, where, the first column plots the same randomly generated CIFAR-10 images as in Figure 16 and the remaining entries in each row are the 14 nearest neighbours (in terms of Euclidean distance) from the training split based on features extracted using a pre-trained Inception net, trained on Imagenet dataset. Figure 18 and 19 presents pixel based and inception feature based training neighbours of some randomly generated images for CELEBA datasets. For both of these datasets, the generated images are visually significantly different as compared to the nearest training examples. This confirms that FlexAE has not memorised the training examples and generates unique, unseen images.

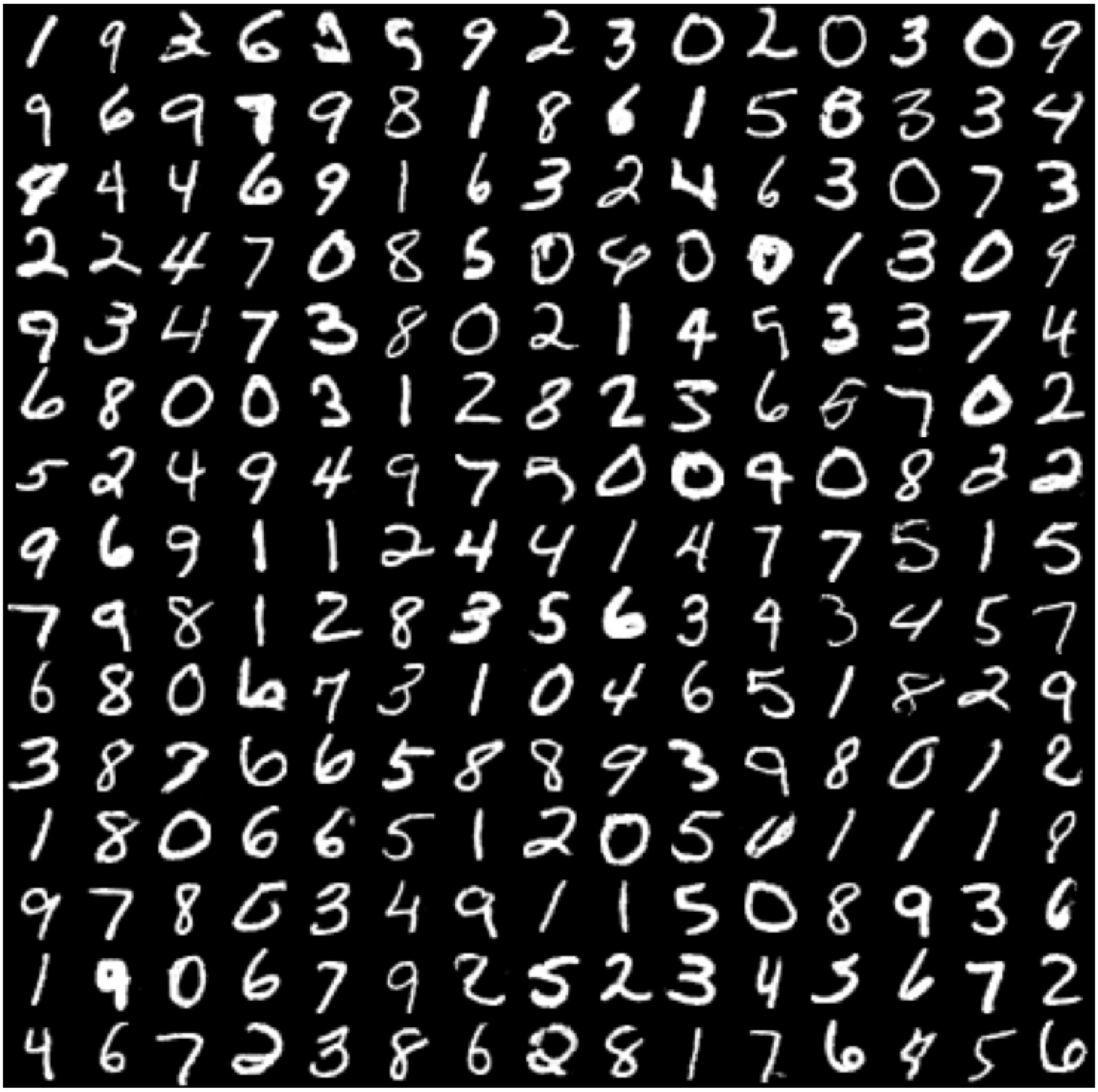


Figure 6: 225 randomly generated MNIST samples using FlexAE-SR.

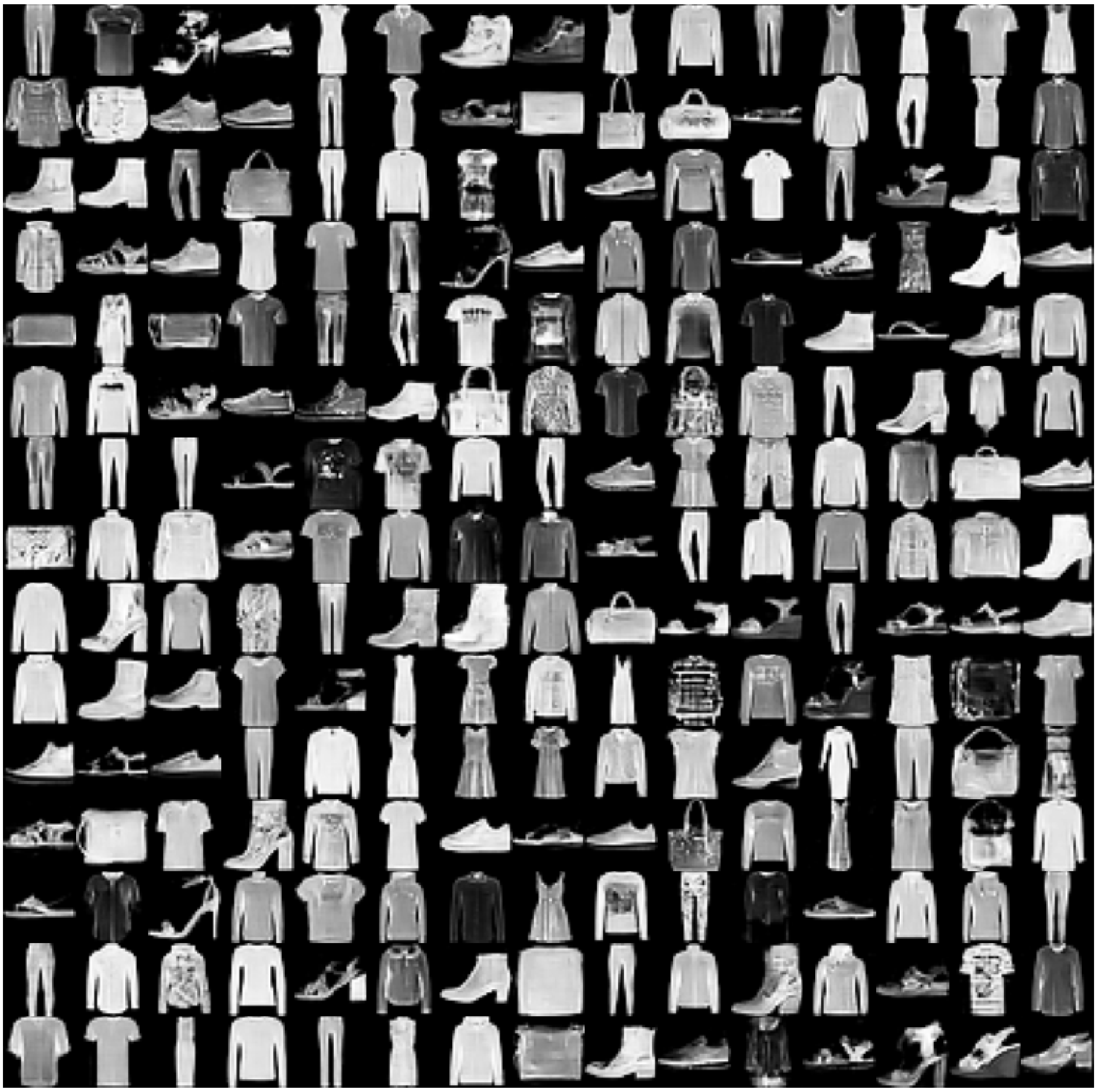


Figure 7: 225 randomly generated Fashion MNIST samples using FlexAE-SR.

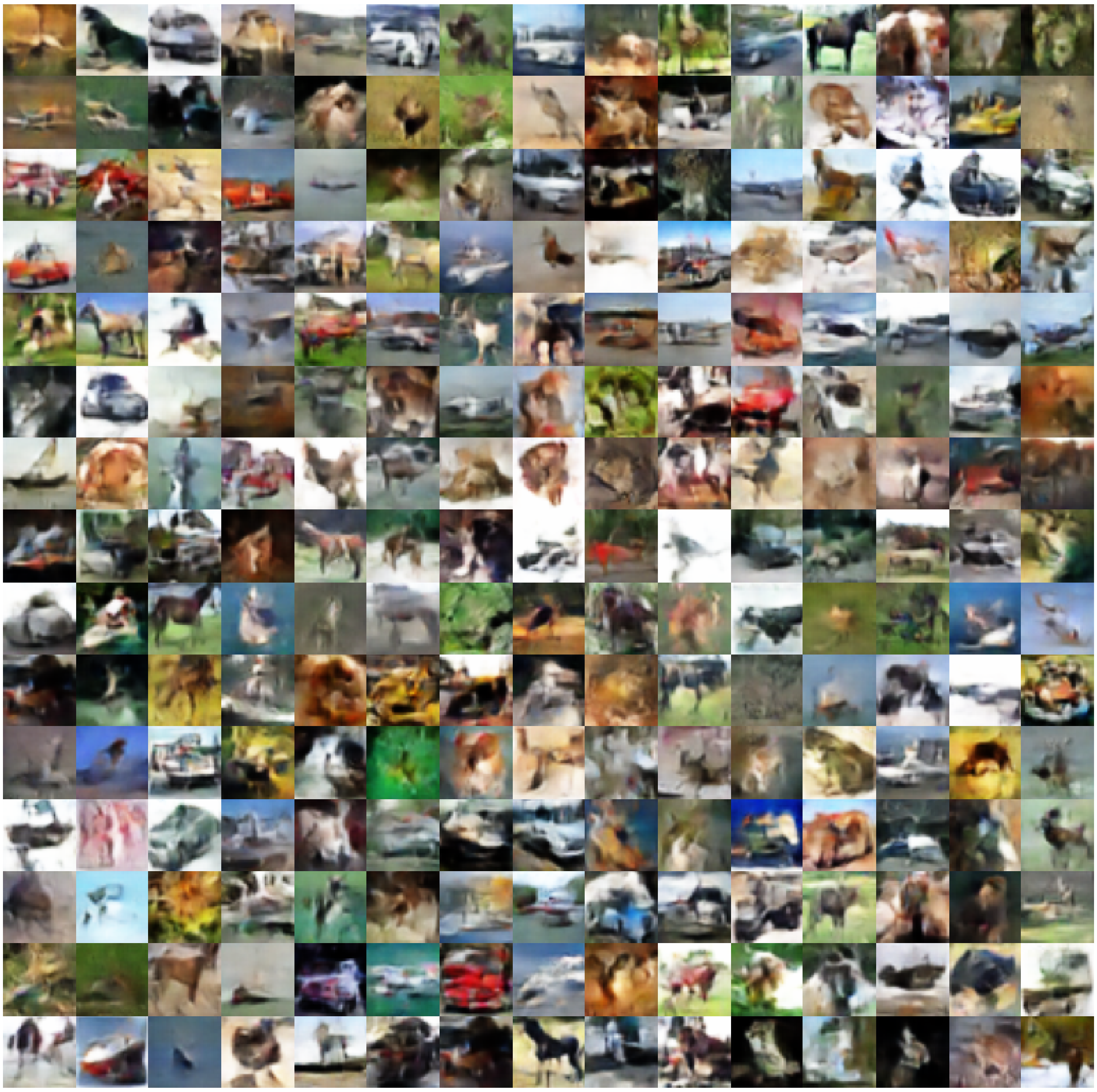


Figure 8: 225 randomly generated CIFAR-10 samples using FlexAE-SR.

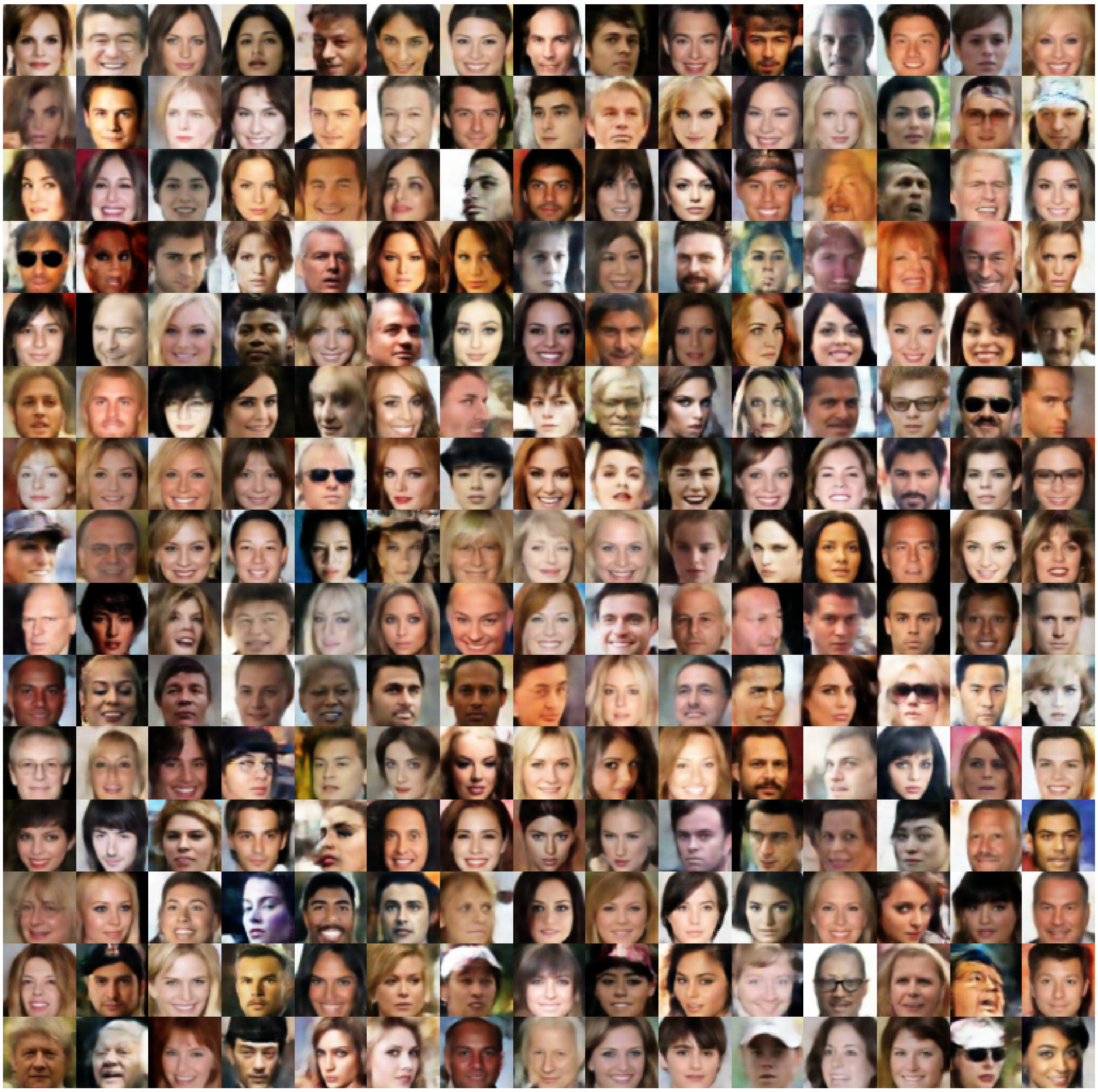
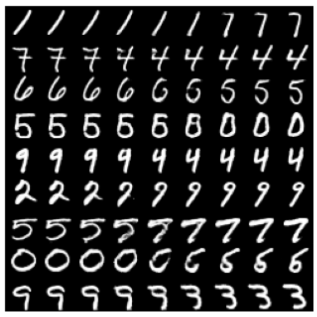
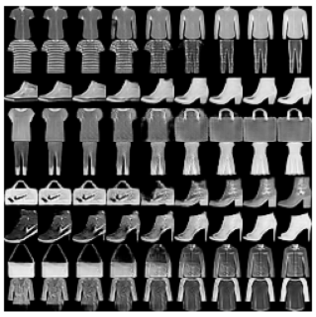
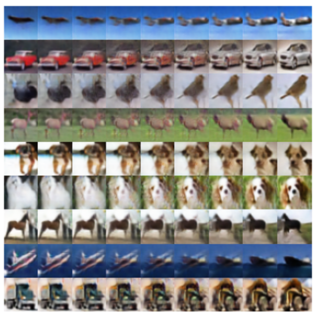


Figure 9: 225 randomly generated CELEBA samples using FlexAE-SR.


(a)


(b)


(c)

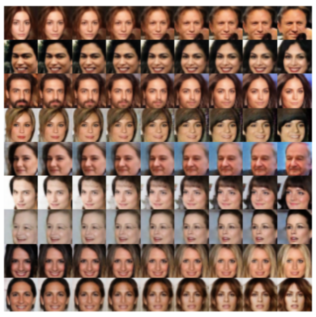
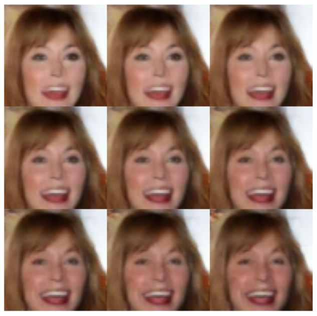
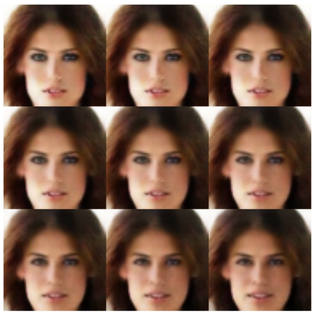
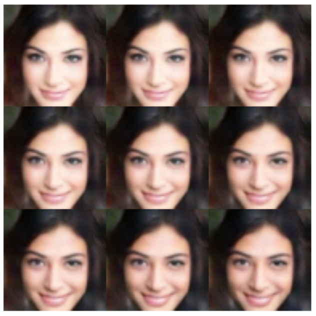

(d)

Figure 10: Interpolations in the latent space of FlexAE-SR on (a) MNIST, (b) Fashion MNIST, (c) CIFAR10 and (d) CelebA. The first and last entry in each row is the reconstructed image corresponding to two real samples. The remaining images are the images obtained by decoding different convex combinations of the latent codes corresponding to the two real images. The central image in each row is decoded from the average latent code of the two real image latent codes. The interpolated images are mostly realistic indicating the smoothness of the FlexAE-SR's latent space.


(a)


(b)


(c)

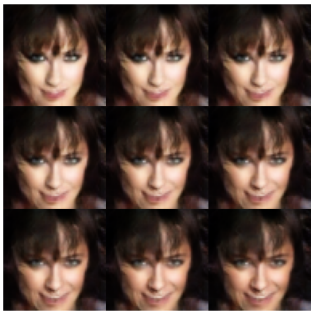
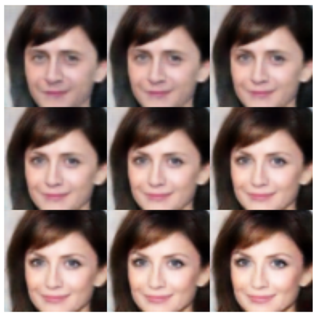
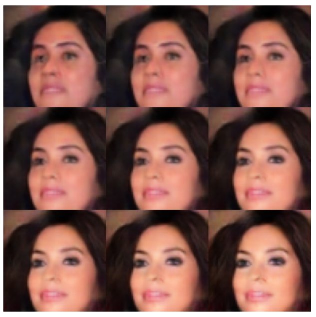
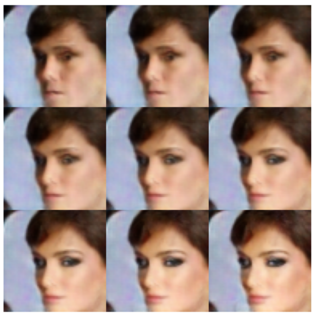

(d)

Figure 11: Interpolations in the latent space of FlexAE-SR on CelebA. Each row in (a) and (b) presents manipulation of the attribute "Big Nose". The central image of each grid in (a), and (b) is a true image without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image with the attribute. Interpolation direction: Top to Bottom, Left to right, like reading a english text book.


(a)


(b)


(c)

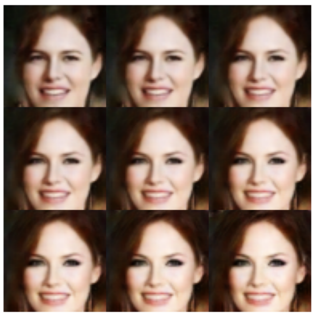

(d)

Figure 12: Interpolations in the latent space of FlexAE-SR on CelebA. Each row in (a) and (b) presents manipulation of the attribute "Heavy Makeup". The central image of each grid in (a), and (b) is a true image without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image with the attribute. Interpolation direction: Top to Bottom, Left to right, like reading a english text book.

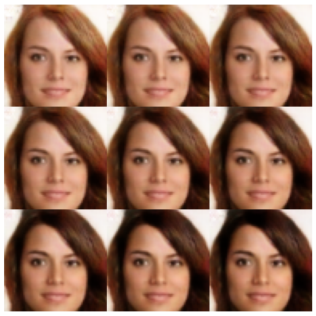
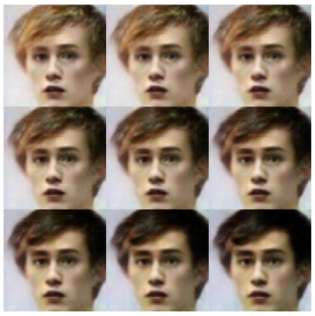
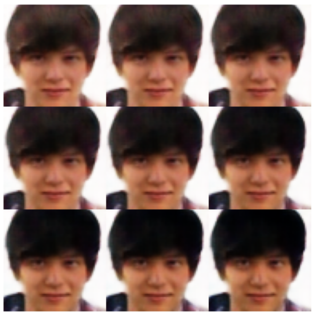
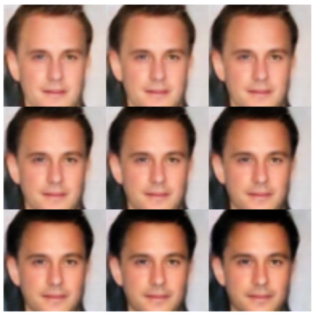

(a)

(b)

(c)

(d)

Figure 13: Interpolations in the latent space of FlexAE-SR on CelebA. Each row in (a) and (b) presents manipulation of the attribute “Black Hair”. The central image of each grid in (a), and (b) is a true image without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image with the attribute. Interpolation direction: Top to Bottom, Left to right, like reading a english text book.

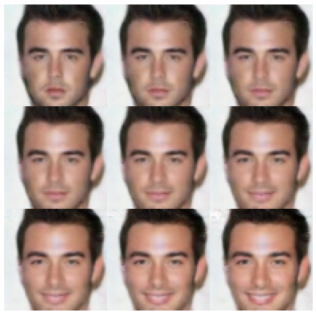
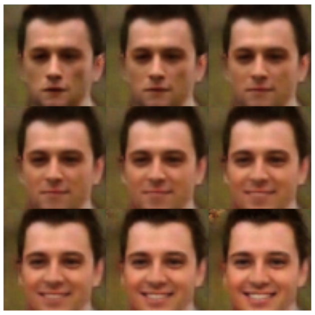
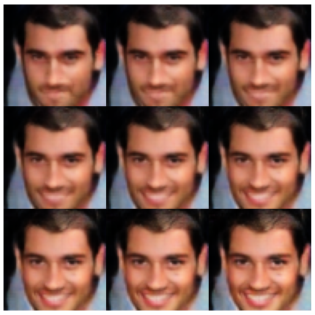
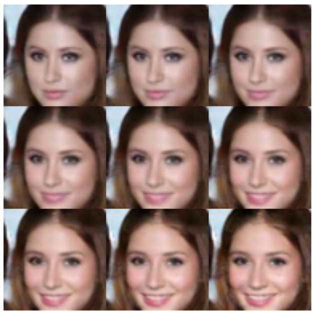

(a)

(b)

(c)

(d)

Figure 14: Interpolations in the latent space of FlexAE-SR on CelebA. Each row in (a) and (b) presents manipulation of the attribute “Smiling”. The central image of each grid in (a), and (b) is a true image without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image with the attribute. Interpolation direction: Top to Bottom, Left to right, like reading a english text book.

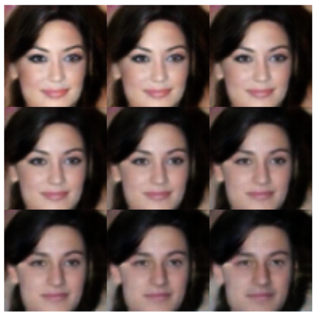
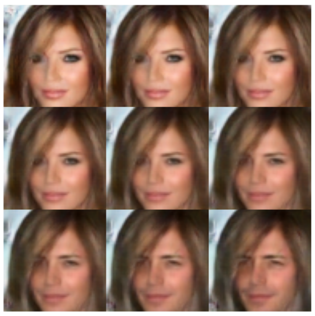
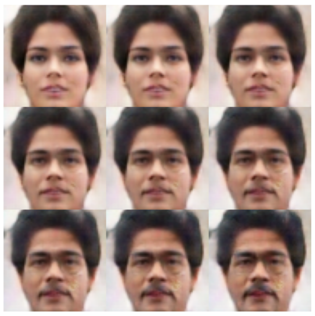
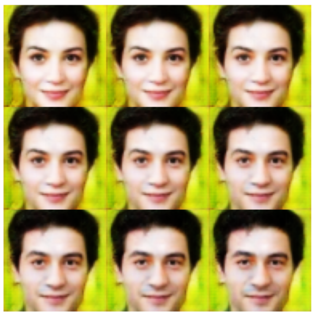

(a)

(b)

(c)

(d)

Figure 15: Interpolations in the latent space of FlexAE-SR on CelebA. Each row in (a) and (b) presents manipulation of the attribute “Male”. The central image of each grid in (a), and (b) is a true image without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image with the attribute. Interpolation direction: Top to Bottom, Left to right, like reading a english text book.

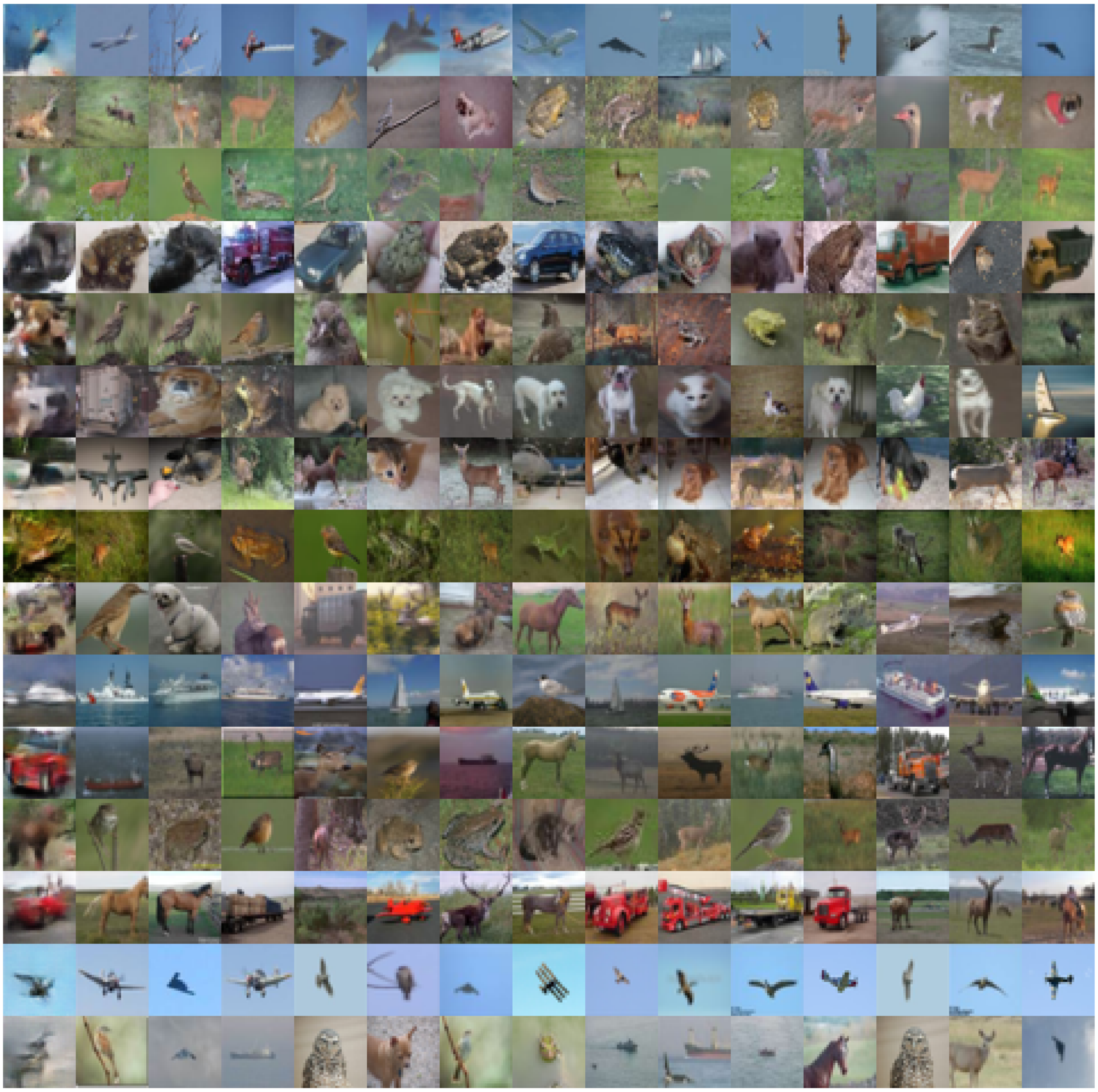


Figure 16: The first entry in each row represents a randomly generated CIFAR-10 object using FlexAE-SR. The remaining entries in each row represents 14 nearest neighbours (in terms of Euclidean distance) from the train split of CIFAR-10 dataset based on pixel values. It is seen that the generated images using FlexAE are very different as compared to the training examples. This confirms that the state of the art FID score and precision recall score obtained using FlexAE is not due to mere overfitting on the training split.

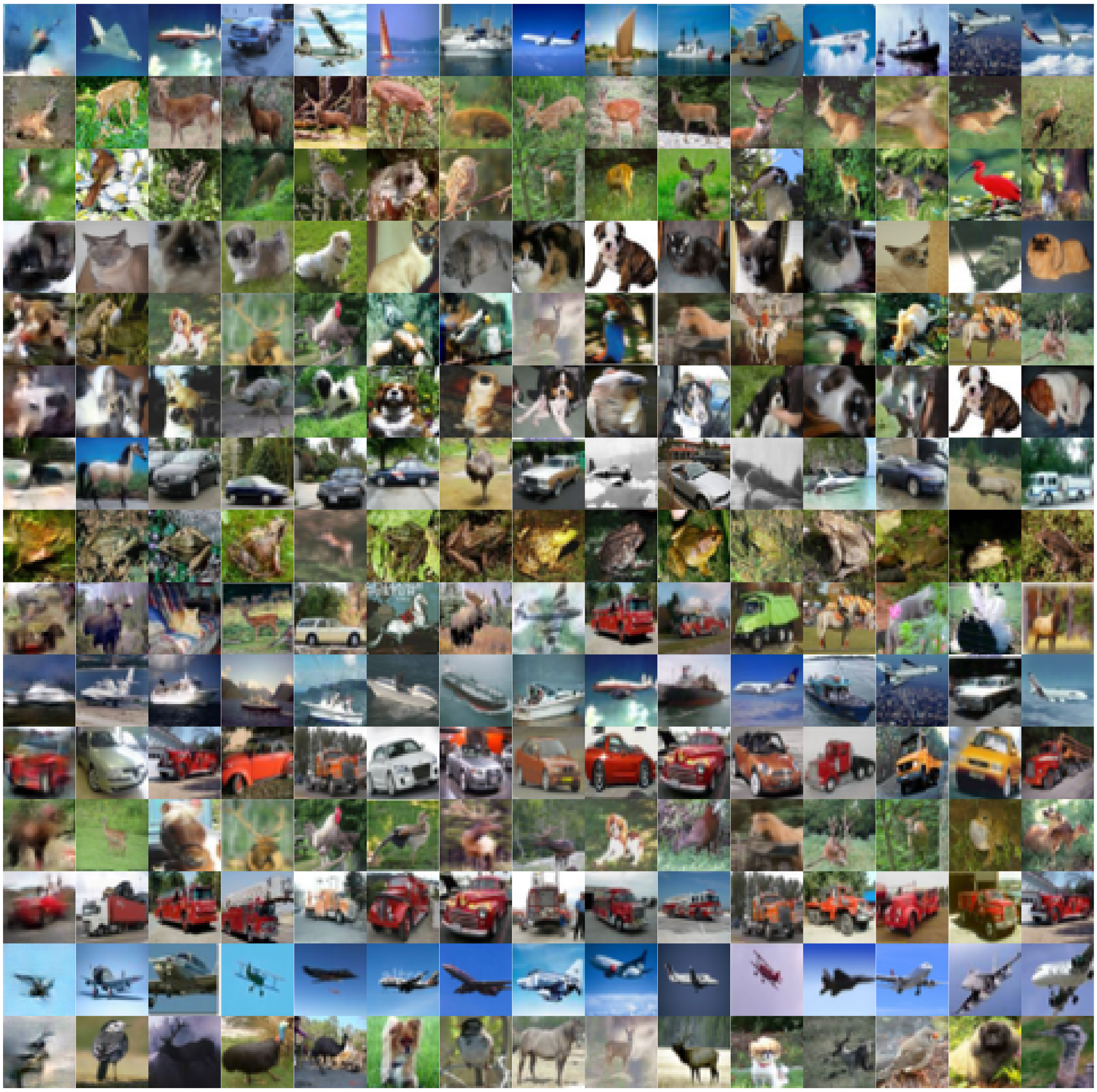


Figure 17: The first entry in each row represents a randomly generated CIFAR-10 object using FlexAE-SR. The remaining entries in each row represents 14 nearest neighbours (in terms of Euclidean distance) from the train split of CIFAR-10 dataset based on features extracted using a pre-trained Inception net. It is seen that the generated images using FlexAE are very different as compared to the training examples. This confirms that the state of the art FID score and precision recall score obtained using FlexAE is not due to mere overfitting on the training split.

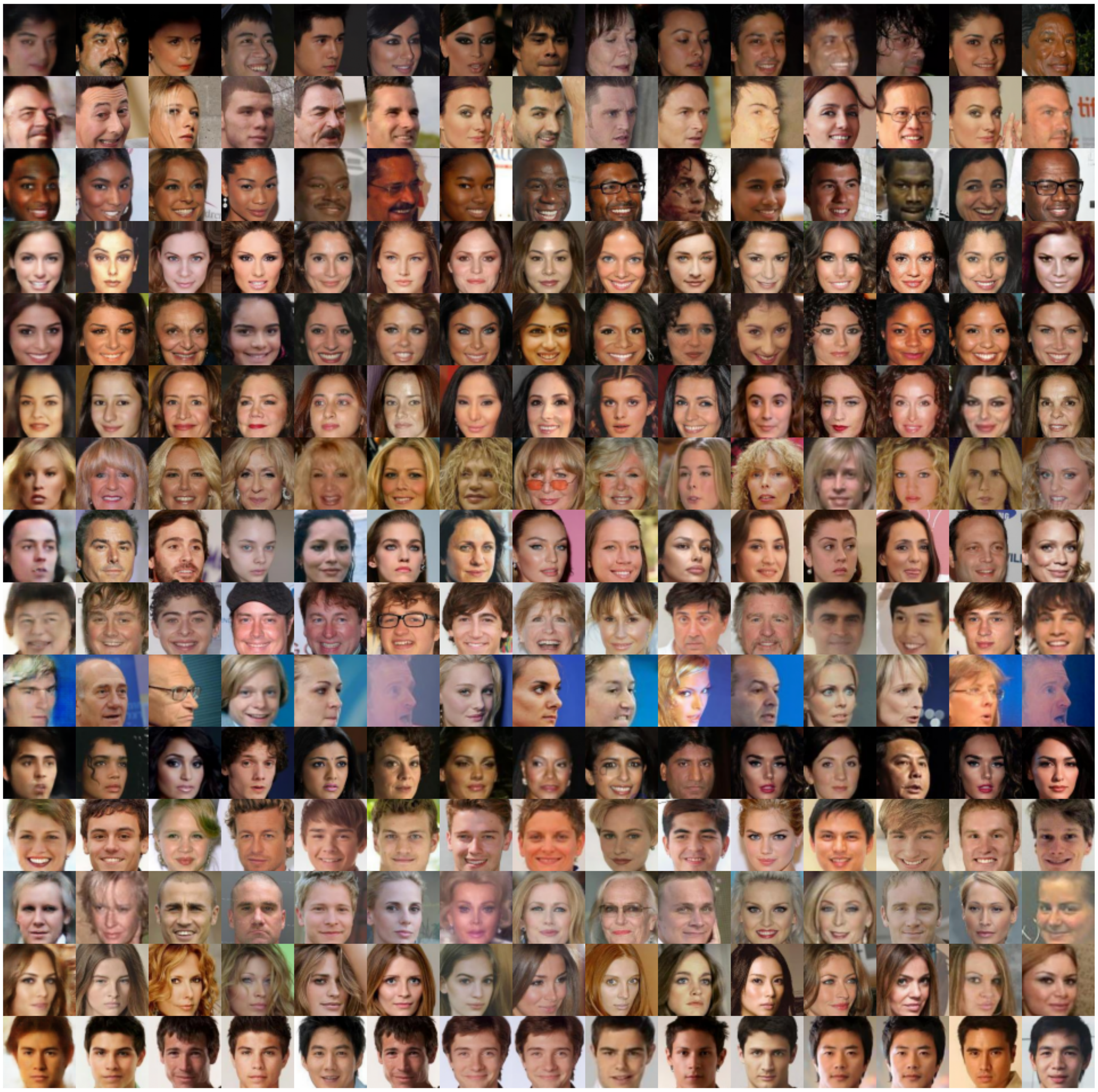


Figure 18: The first entry in each row represents a randomly generated face using FlexAE-SR. The remaining entries in each row represents 14 nearest neighbours (in terms of Euclidean distance) from the train split of CELEBA dataset based on pixel values. It is seen that the generated images using FlexAE are very different as compared to the training examples. This confirms that the state of the art FID score and precision recall score obtained using FlexAE is not due to mere overfitting on the training split.

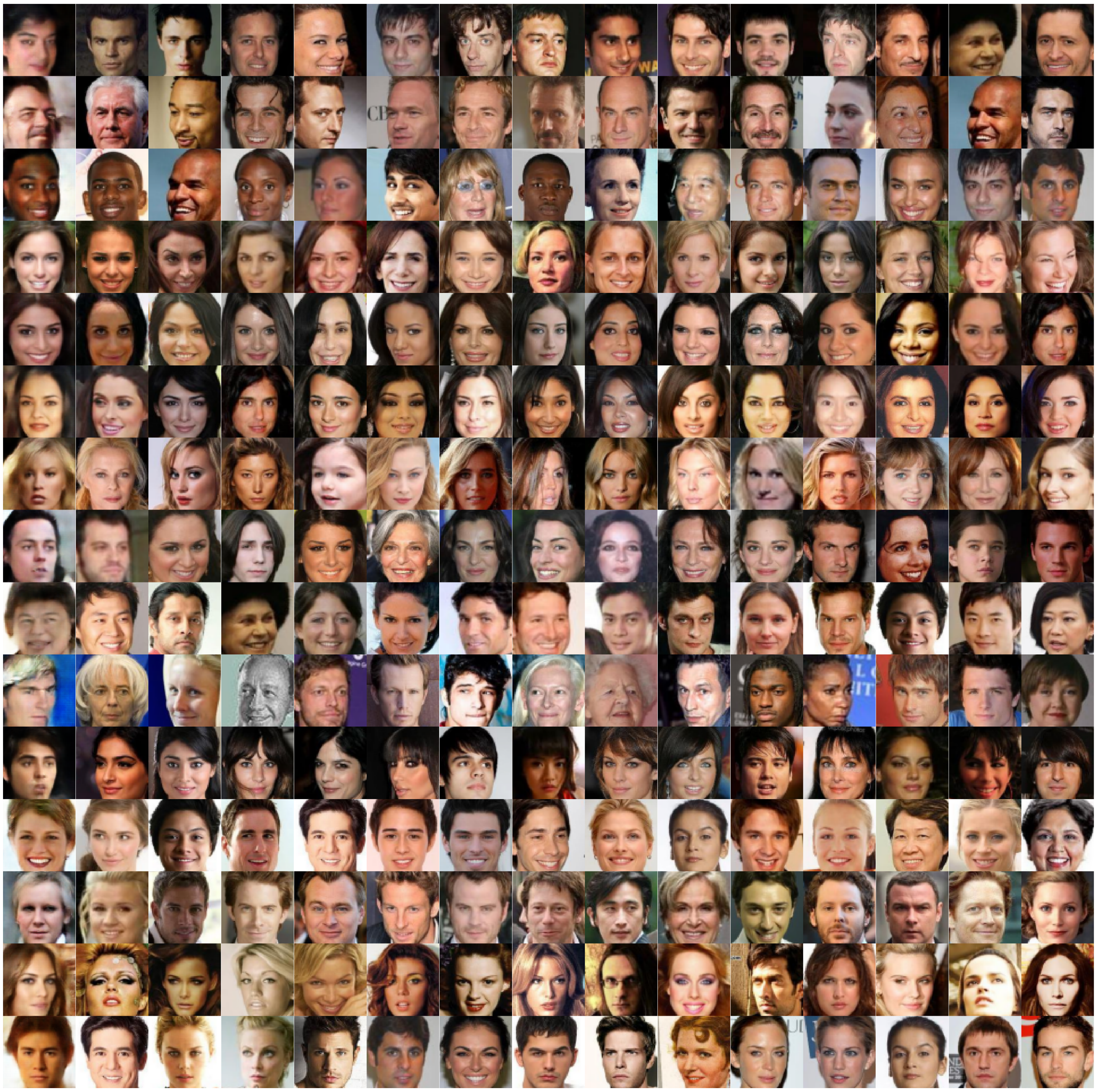


Figure 19: The first entry in each row represents a randomly generated face using FlexAE-SR. The remaining entries in each row represents 14 nearest neighbours (in terms of Euclidean distance) from the train split of CELEBA dataset based on features extracted using a pre-trained Inception net. It is seen that the generated images using FlexAE are very different as compared to the training examples. This confirms that the state of the art FID score and precision recall score obtained using FlexAE is not due to mere overfitting on the training split.