To Regularize or Not To Regularize? The Bias Variance Trade-off in Regularized AEs

Arnab Kumar Mondal  
IIT Delhi  
anz188380@cse.iitd.ac.in

Himanshu Asnani  
TIFR Mumbai  
himanshu.asnani@tifr.res.in

Parag Singla  
IIT Delhi  
parags@cse.iitd.ac.in

Prathosh AP  
IIT Delhi  
prathoshap@ee.iitd.ac.in

Abstract

Regularized Auto-Encoders (AE) form a rich class of methods within the landscape of neural generative models. They effectively model the joint-distribution between the data and a latent space using an Encoder-Decoder combination, with regularization imposed in terms of a prior over the latent space. Despite their advantages such as stability in training, the performance of AE based models has not reached that of the other models such as GANs. While several reasons including the presence of conflicting terms in the objective, distributional choices imposed on the Encoder and the Decoder, and dimensionality of the latent space have been identified as possible causes for the suboptimal performance, the role of the regularization (prior distribution) imposed has not been studied systematically. Motivated by this, we examine the effect of the latent prior on the generation quality of the AE models in this paper. We show that there is no single fixed prior which is optimal for all data distributions, given a Gaussian Decoder. Further, with finite data, we show that there exists a bias-variance trade-off that comes with prior imposition. As a remedy, we optimize a generalized ELBO objective, with an additional state space over the latent prior. We implicitly learn this flexible prior jointly with the AE training using an adversarial learning technique, which facilitates operation on different points of the bias-variance curve. Our experiments on multiple datasets show that the proposed method is the new state-of-the-art for AE based generative models.

1 Introduction

Auto-Encoder (AE) based latent variable models form a major class among modern neural generative models. Variational Auto-Encoders (VAEs) [1], Beta-VAE [2], Adversarial Auto-Encoders (AAEs) [3], Wasserstein Auto-Encoders (WAEs) [4], Regularized Auto-Encoders [5], 2-Stage VAE [6], VAE with hierarchical priors [7] and MaskAAE [8] are a few examples from this family. They implicitly define a joint distribution over the input data and a lower-dimensional latent space. They do so by approximating the true posterior of the latent conditioned on the data, using a variational distribution, parameterized using a Neural network called the Encoder. Subsequently, a Decoder network is used to conditionally sample from the data distribution given a sample from the latent distribution. The parameters of the Encoder and the Decoder networks are learned by optimizing a lower-bound on the data likelihood. The framework of AE-based generative models are attractive because of their 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 [6].

Several aspects such as the loss function used for optimization [2][9], presence of conflicting terms in the optimization objective [10][11], distributional choices (E.g., Gaussianity) imposed on the Encoder and Decoder [12][13], dimensionality of the latent space used [6][8], the mismatch between the learned and imposed prior [11][14] have been identified as possible causes for the sub-optimal performance.
of the AE-based models. Many remedial measures including modification of the objective function \cite{12, 2, 15}, use of non-Gaussian Encoder/Decoder \cite{9, 16, 4}, masking of spurious latent dimensions \cite{8, 6}, incorporating a richer class of priors on latent space \cite{14, 17, 7} have been proposed in the literature to address some of these issues. While these modifications have improved AE models’ performance, they are still far behind SoTA generative models \cite{6, 8}.

Motivated by above, we examine the effect of one of the critical components of the AE-based - the prior distribution (or the type of regularization) that is imposed on the latent space. It is well-recognized that the aggregated distribution learned by the Encoders of the AE-model often fails to match the assumed latent prior leading to sub optimal performances \cite{11, 14, 6}. While there have been some attempts towards addressing this problem by allowing richer class of priors \cite{12, 7}, there is no justification for the specific class of priors used. In fact, we show that making assumptions on the prior oblivious to the data can lead to sub optimal results (See Fig. 1). On the other hand, some papers have argued for altogether getting rid of the prior over the latent \cite{5, 11}; however, here a natural question arises whether this will likely lead to overfitting, especially with finite data. Hence, we believe that a more systematic treatment of the bias-variance trade-off associated with the choice of the latent prior for AE based models is warranted. Specifically, the contributions of our work are listed below:

1. We formulate an optimization problem with an additional state-space involving a learnable latent prior distribution and show the existence of its optimality. We also derive the necessary and sufficient conditions to be satisfied to achieve this optimal value.
2. We argue against having a fixed prior by showing that the necessary conditions are not satisfied under simplistic Gaussian Decoders and unimodal priors.
3. We show that, with finite data and model capacities, there exists a bias-variance trade off between the generation quality and the choice of imposed latent prior.
4. We propose a model (called the FlexAE) that can impose flexible learnable priors facilitating the trade off between the bias and variance on-the-go during AE-training.
5. We empirically demonstrate our claims through extensive experimentation on synthetic and real-world datasets by achieving significant improvement over the SoTA AE models.

2 Background and Related Work

The general theme in AE-based generative models is to maximize a lower bound (Evidence Lower Bound or ELBO) on the data likelihood. ELBO typically comprises of two terms - (i) conditional data likelihood under a variational latent posterior and (ii) KL divergence between the variational latent posterior and the latent prior. Variational Autoencoders (VAE) \cite{1} 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) \cite{3} and Wasserstein Auto-Encoders (WAEs) \cite{4} where different divergence metrics such as Jensen-Shannon divergence and Wasserstein distance respectively, are used to bring the aggregated posterior close to the assumed latent prior.

Even though VAE (and related models) provides a solid framework for AE-based generative models, several drawbacks are associated with it. It is shown that there exists a conflict between the two terms of the ELBO \cite{2, 11, 13}. A few remedial measures such as introduction of a tunable parameter in the second term of the ELBO \cite{10}, use of additional penalties such as mutual information \cite{12}, total correlation \cite{13}, and altering the optimization procedure \cite{13} have been proposed.

Another issue with VAE (and associated AE-models) is the simplistic distributional choices made for Encoder/Decoder networks \cite{18}. For instance, it is a usual practice to use the mean squared error loss as a proxy for the first term in ELBO which is operationally equivalent to using a Gaussian Decoder. Many papers have addressed this issue by employing richer class of distributional choices for Encoder/Decoder models. \cite{19} implements a Bayesian nonparametric version of the variational autoencoder that has a latent representation with stochastic dimensionality, that could represent richer class of distributions. Invertible flow-based generative models \cite{20, 21} capitalize on the idea of normalizing flow for the Encoder and Decoder networks. VAE/GAN \cite{9}, VGH/VGH++ \cite{18} incorporates adversarial loss for distributional matching at the Decoder.
Another important issue is the distributional choice made on the latent prior. It has been shown that even with a simplistic Gaussian prior, a mismatch between the true and the model latent dimensions leads to a bad performance \[8\]. This issue is addressed in \[6\] by using 2-stages of VAEs, one on the data space and the second of the latent space. MaskAAE \[8\] explicitly masks the spurious latent dimensions via a learnable binary mask with adversarial learning. Furthermore, it is shown that the often-occurring mismatch between the aggregated variational posterior and the latent prior results in bad generation quality. This is ascribed to the usage of a simplistic Gaussian prior on the latent space. Several papers try to alleviate this problem, broadly in two ways (i) using a richer class of parametric priors on the latent space \[14\] \[7\] \[22\] and (ii) using a post-hoc technique to minimize the KL-divergence or sample from the latent space without regularizing it \[23\] \[5\] \[17\].

Among the first category of methods, VamPrior \[14\] assume the prior to be a mixture of the conditional distributions. Another important issue is the distributional choice made on the latent prior. It has been shown that the log-likelihood \(\mathcal{L}\) of the data distribution under a model \(p_\theta(x)\) can be written as follows:

\[
\mathcal{L} = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - D_{KL}(q_\phi(z)||p_\psi(z)) - I(x;z_\phi) + \mathbb{E}_{p_d(x)} \left[ D_{KL}(q_\phi(z|x)||p_\psi(z|x)) \right]
\]

(1)

Here \(I(\cdot)\) denotes the mutual information, \(D_{KL}\) is the Kullback-Leibler divergence. \(q_\phi(z)\) denotes the distribution of the latent encoded by the encoder \(E_\phi\) of the AE, also called as the aggregated posterior, defined as \(q_\phi(z) = \int q_\phi(z|x)p_d(x)dx\). \(p_\phi(z)\) is the prior distribution (or regularization) that is imposed on the encoded latent space. Eq. (1) follows from the break down of the joint distribution at the Decoder as \(p_\theta(x,z) = p_\theta(x|z)p_\phi(z)\) (cf. Sec. 6.1 of the supplementary material for breakup details). We define the aggregation of the terms (I), (II), and (III) as the Evidence Lower Bound \(ELBO(\theta, \phi, \psi)\):

\[
ELBO(\theta, \phi, \psi) = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - D_{KL}(q_\phi(z)||p_\psi(z)) - I(x;z_\phi)
\]

(2)

and denote the term (IV) as \(\Delta(\theta, \phi) \geq 0\) implying \(LLE(\theta) = ELBO(\theta, \phi, \psi) + \Delta(\theta, \phi)\). In the case of classical AE-models such as VAEs, AAEs etc., \(\psi\) is kept a constant \(\psi_0\) which corresponds to...
standard normal distribution; \( p_{\psi_0} \sim \mathcal{N}(0, I) \). Hence, in such models, the optimization is only over the Encoder and Decoder parameters, while in a general setup (such as ours), \( \psi \) is also trainable. The goal of an AE-model is to optimize for these parameters by optimizing ELBO. With these, we provide the necessary and sufficient conditions for the optima of the ELBO to be achieved, below.

**Theorem 1.** Let the optimal parameters which maximize the ELBO term be \((\phi^*, \theta^*, \psi^*) = \arg \max_{\phi, \theta, \psi} \text{ELBO}(\theta, \phi, \psi)\). Then the following necessary and sufficient conditions hold:

\[
\begin{align*}
C1: & \quad p_{\psi^*}(z) = q_{\phi^*}(z) \\
C2: & \quad p_{d}(x)q_{\phi^*}(z|x) = q_{\phi^*}(z)p_{\theta^*}(x|z) \\
C3: & \quad \text{ELBO}(\phi^*, \theta^*, \psi^*) = -H(x) \\
C4: & \quad \Delta(\theta^*, \phi^*) = 0
\end{align*}
\]

Here \( H(x) \) denotes the entropy of the data distribution. Please refer to Sec. 6.2 of the Supplementary material for proof.

Theorem 1 asserts that to reach the optimal value (data entropy) of the ELBO the following conditions should be met: (i) the prior (even when it is learnt) that is imposed on the Encoder should match with the aggregated posterior, (ii) the joint distributions learned by the Encoder and the Decoder networks should match. Note that even though C1 has been laid down elsewhere \cite{13, 14}, the rest have not been explicitly noted in the literature in the context of generative AE-models. We believe these conditions are important since they not only establish a relationship between latent and learned priors (C1), and the joint distributions enforced by encoder and decoder (C2), but also show that when C1 and C2 are met, we can achieve the optimum likelihood value (C3 and C4). In the next section, we show that fixing a prior violates the conditions.

### 3.2 Sub-optimality of a Fixed Prior

In this section, we argue against fixing both a Gaussian prior as well as any non-Gaussian prior, *a priori*, by showing the existence of data distributions where such a choice of prior would not be optimum, under two different cases.

**Theorem 2.** If the true data distribution \( p_d(x) \) is not Gaussian, then under the assumption of Gaussian Decoder, \( p_{\theta}(x|z) \sim \mathcal{N}(\mu_\theta, \Sigma_\theta) \) a Gaussian prior, \( q_{\psi_0}(z) \sim \mathcal{N}(0, I) \), an AE-based model cannot reach the optimum value in ELBO maximization and hence cannot maximize the likelihood.

**Theorem 3.** If the true data distribution \( p_d(x) \) is Gaussian, then under the assumption of Gaussian Decoder, \( p_{\theta}(x|z) \sim \mathcal{N}(\mu_\theta, \Sigma_\theta) \) and Gaussian Encoder, \( q_{\phi}(z|x) \sim \mathcal{N}(\mu_\phi, \Sigma_\phi) \), an AE based generative model cannot reach the optimum value in ELBO with a non Gaussian prior \( q_{\psi}(z) \).

Proofs of Theorem 2 and 3 are provided in the supplementary material Sec. 6.2. Note that similar arguments can be made for any fixed unimodal prior distribution with Gaussian Decoders. Even though previous literature has discussed about the possible insufficiency of the Gaussian prior \cite{6, 10}, Theorem 2 is a first formal statement to this effect. Further, Theorem 3 is particularly interesting because despite the arguments favouring more expressive priors \cite{7, 14}, it counterintuitively shows insufficiency of a more sophisticated non-Gaussian prior when data is normally distributed. In summary, Theorem 2 and 3 collectively assert that there exists data distributions (and AE-models) for which fixing any kind of prior oblivious to that data can be detrimental.

To illustrate these, we train three AE-models (VAE with Gaussian prior, WAE with GMML prior, and FlexAE with learnable prior) with same Encoder/Decoder architectures on two synthetic datasets. In the first dataset, a latent space is sampled from a 2D six-component GMM which is then passed through an MLP to generate 128-dimensional data points (cf. Sec. 7.4 and 8.1 of Suppl. for details). In the second dataset, a 2D unimodal Gaussian latent space is linearly transformed to generate data (which will also be Gaussian). The t-SNE plot of the data and the learned latent space \( (q_{\phi}(z)) \) of all models along with the Fréchet Distance (FD) \cite{25} is computed on the generated data (lower the better) is presented in Fig. 1. For the first dataset through multimodal distribution, it is seen that models with a fixed multimodal latent prior (WAE-GMM) and with flexible prior (FlexAE, ours), retain the latent structure and thus offer better FD on the generated data (Theorem 2). Whereas for the second dataset with Gaussian distribution, the model with Gaussian prior (VAE) and flexible prior (FlexAE, ours) perform well, whereas WAE-GMM performs worse (Theorem 3). It is interesting to note that, despite
Figure 1: Visualization of data (t-SNE) and latent spaces for the synthetic data. (a), (f): t-SNE of the true data space, (a) Non-Gaussian, (f) Gaussian; (b), (g): True Latent space; (c), (h): latent space learned by the VAE [1]; (d), (i): latent space learned by the WAE [4] with GMM prior; and (e), (j): latent space learned by the proposed FlexAE model, along with generation Fréchet Distance (FD) in each case. For multimodal data (row 1), model with multimodal prior (WAE-GMM) and FlexAE perform better (Theorem 2) and for Gaussian data (row 2), model with unimodal latent prior (VAE) and FlexAE perform better (Theorem 3). In both cases, the latent space learned by FlexAE (ours) with a learnable prior, is very close to the true latent, yielding best generation quality overall.

the exact same architectural settings in both the cases, the latent space learned by FlexAE closely follows that of the true latent.

3.3 Choosing the “Right” Prior: The Bias-variance Trade-off

From the previous discussion, it is clear that any fixed prior is not optimal given an arbitrary data distribution. In this section, we first show that conditions in Theorem 1 will be naturally satisfied without imposition of any prior, provided there is access to infinite data and Encoder/Decoder networks are of sufficient capacity. However, subsequently, we argue that in practical conditions, with finite data, there exists a bias-variance trade-off associated with choosing a prior.

**Theorem 4.** With sufficient capacity Encoders and Decoders, the AE-model without any prior or regularization on the latent space will achieve the optimality defined in Theorem 1.

Proof can be found in Sec. 6.2 of the supplementary material. Theorem 4 suggests that one can have sufficiently capable Encoder and Decoder networks and do away with any prior on the latent space. This however demands sampling from $p_\psi(z)$ to facilitate data generation, via Decoder. This could be fulfilled using a post facto sampler such as GMM, Markov-Chain Monte-Carlo (MCMC) or a GAN learned on the trained latent space of the AE. Even though this is possible in theory, we argue that it is not an optimal procedure in practice because of the limited data availability and finite capacity of models. Specifically, it is easy to show that with finite data, a trivial solution can satisfy all of the conditions in Theorem 1. Specifically, $q_{\phi}(z|x_i)$ would be Dirac-delta functions at all input data points $x_i$. Subsequently, the post-hoc sampler (E.g., GAN) will learn to sample from finite set of Dirac-deltas. This would lead to over fitting and poor generalization. On the other hand, as shown in Theorem 2 and 3, a "wrong" prior (or regularization) will also impact the generation quality. This is the infamous bias-variance trade-off associated with choosing a prior.

3.4 The Proposed Model: FlexAE

Our model, called the Flexible AE or FlexAE, consists of an Auto-Encoder and a Wasserstein Generative Adversarial Network (WGAN) [27] on its latent space that parameterizes the latent prior
We demonstrate the efficacy of FlexAE on three real-world datasets: MNIST [32], CIFAR-10 [33], and CelebA [34]. We perform three sets of experiments to demonstrate different claims made.

4 Experiments and Results

We demonstrate the efficacy of FlexAE on three real-world datasets: MNIST [32], CIFAR-10 [33], and CelebA [34]. We perform three sets of experiments to demonstrate different claims made.

4.1 Baseline Experiments, Comparison with State-of-the-Art

Methodology: The first task is to evaluate the FlexAE as a generative model. Owing to the heterogeneity across different class of generative frameworks [35][36], there is not a universally accepted metric for validating the performance of generative models [35][37]. However, Fréchet Inception Distance (FID) [25] is one of the most commonly used evaluation methods as it correlates well with human visual perception [36]. However, as observed in [37], 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 [37] 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 [1], β-VAE [2], VAE-VamPrior [14], VAE-IOP [17], WAE [4], a plain with AE post-hoc GMM, RAE+GMM.
with other AE-based generative models. It is seen that while models with parametric learnable priors
This confirms our hypothesis of existence of a Bias-Variance curve. Please note, in Experiment 1,
terms of the parameters of P-GEN.
architectural choice for the P-GEN imposes a bias, the flexibility (needed for trade off) is ensured in
signalling over fitting. A reverse observation could be made about the high-bias low capacity models.
Table 3 shows that there is a performance drop at the either sides of Model 3. As the
capacity of the P-GEN increases, the reconstruction FID decreases while generation FID increases,
apparent. Models of huge capacity are needed to observe similar effects of the entire dataset.
Table 1 compares the reconstruction and generation FID scores (lower is better) of FlexAE
high capacity (details of models in Sec. 8.2, Table 8 of the supplementary material), on a small subset
of training data (5000 samples). Sub-sampling is to ensure that effect of bias-variance trade-off is
impact (2SV AE, MaskAAE). A relatively better performance of RAE+GMM, InjFlow shows that
Results: Table 1 compares the reconstruction and generation FID scores (lower is better) of FlexAE
and its performance on MNIST and CelebA are comparable to that of the GANs. A similar trend
offers the best performance on all three datasets as compared to other AE based generative models
Recalling confirming its effectiveness in generating samples that are of both high quality and variety.

Table 1: Comparison of FID scores [25] on real datasets. Lower is better.

|                  | MNIST | CIFAR10 | CELEBA |
|------------------|-------|---------|--------|
|                  | Rec.  | Gen.    | Rec.   | Gen.    |
| VAE [1]          | 65.10 | 57.04   | 176.5  | 169.1   | 62.36  | 72.48  |
| β-VAE [2]        | 7.91  | 24.31   | 43.86  | 83.59   | 30.06  | 50.66  |
| VAE-Vamprior [14]| 11.01 | 49.75   | 107.33 | 161.02  | 49.71  | 64.26  |
| VAE-IOP [17]     | 8.01  | 32.61   | 92.17  | 141.92  | 41.52  | 57.30  |
| WAE-GAN [4]      | 8.06  | 13.30   | 42.39  | 72.90   | 29.34  | 39.58  |
| AE + GMM (L2) [5]| 8.09  | 12.14   | 41.45  | 70.97   | 30.16  | 43.89  |
| RAE + GMM (L2) [5]| 6.15  | 7.30    | 40.48  | 69.24   | 29.05  | 35.30  |
| VAE + FLOW [20]  | 8.62  | 20.17   | 43.87  | 73.28   | 36.31  | 42.39  |
| InjFlow [22]     | 7.40  | 35.96   | 40.11  | 78.78   | 27.93  | 47.70  |
| VAE + (L2) [5]   | 7.40  | 9.93    | 40.11  | 68.26   | 27.93  | 40.23  |
| MaskAAE [8]      | 8.46  | 10.52   | 58.40  | 71.90   | 35.75  | 40.49  |
| FlexAE (Proposed)| **4.33** | **4.69** | **39.91** | **62.66** | **20.47** | **24.72** |

[5], VAE+Flow [20], InjFlow [22], 2-stage VAE [6] and MaskAAE [8], with same architectures.

Results: Table 1 compares the reconstruction and generation FID scores (lower is better) of FlexAE with other AE-based generative models. It is seen that while models with parametric learnable priors (vamprior, IOP, Flow) offer some improvement over the naive VAE, they are far from being optimum. It is also seen that complex prior models tend to over fit more (gap between the generation and reconstruction FIDs). Further, having the “right” dimensional latent space seems to have significant impact (2SVAE, MaskAAE). A relatively better performance of RAE+GMM, InjFlow shows that while absence of prior imposition will reduce the bias, it might lead to over fitting. Finally, FlexAE offers the best performance on all three datasets as compared to other AE based generative models and its performance on MNIST and CelebA are comparable to that of the GANs. A similar trend is observed with the Precision/Recall numbers in Table 2 (We only use better SoTA models for comparison). It is seen that FlexAE offers significantly better numbers in terms of both Precision and Recall confirming its effectiveness in generating samples that are of both high quality and variety.

Table 2: Comparison of Precision/Recall scores [27] on real datasets. Higher is better.

|                  | MNIST | CIFAR10 | CELEBA |
|------------------|-------|---------|--------|
|                  |       |         |        |
| VAE [1]          | 0.69/0.76 | 0.23/0.47 | 0.47/0.58 |
| 2S-VAE [6]       | 0.97/0.98 | 0.47/0.76 | 0.75/0.72 |
| RAE + GMM (L2) [5]| 0.98/0.98 | 0.61/0.87 | 0.74/0.75 |
| MaskAAE [8]      | 0.94/0.96 | 0.58/0.83 | 0.59/0.68 |
| FlexAE (Proposed)| **0.99/0.99** | **0.68/0.85** | **0.89/0.88** |

4.2 Bias-Variance Trade-off

Methodology: To evaluate our claims on the Bias-Variance trade-off, we repeat the generation experiments by varying the capacity of the prior generator (P-GEN) from very low capacity to very high capacity (details of models in Sec. 8.2, Table 8 of the supplementary material), on a small subset of training data (5000 samples). Sub-sampling is to ensure that effect of bias-variance trade-off is apparent. Models of huge capacity are needed to observe similar effects of the entire dataset.

Results: Table 3 shows that there is a performance drop at the either sides of Model 3. As the capacity of the P-GEN increases, the reconstruction FID decreases while generation FID increases, signalling over fitting. A reverse observation could be made about the high-bias low capacity models. This confirms our hypothesis of existence of a Bias-Variance curve. Please note, in Experiment 1, the architecture of the P-GEN was kept fixed across all datasets. Therefore, even though the mere architectural choice for the P-GEN imposes a bias, the flexibility (needed for trade off) is ensured in terms of the parameters of P-GEN.
Table 3: Variation of reconstruction and generation FID scores on limited training datasets with varying P-GEN capacity, demonstrating bias-variance trade-off. Models (1-6) are presented in increasing order of capacity.

| Dataset   | Model 1  | Model 2  | Model 3  | Model 4  | Model 5  | Model 6  |
|-----------|----------|----------|----------|----------|----------|----------|
|           | Rec.     | Gen.     | Rec.     | Gen.     | Rec.     | Gen.     | Rec.     | Gen.     | Rec.     | Gen.     |
| MNIST     | 60.51    | 55.49    | 21.09    | 53.93    | 13.41    | 42.14    | 14.49    | 31.90    | 8.11     | 63.64    | 8.94     | 62.43    |
| CIFAR-10  | 154.17   | 135.32   | 91.85    | 104.06   | 82.95    | 108.63   | 83.88    | 108.46   | 94.2     | 120.64   | 94.54    | 121.96   |
| CELEBA    | 79.04    | 66.84    | 42.77    | 56.16    | 47.02    | 54.32    | 42.75    | 54.14    | 44.02    | 59.3     | 39.1     | 58.49    |

4.3 Smoothness of the Latent Space

**Methodology:** To ascertain the smoothness of the learned latent space and that FlexAE doesn’t overfit, we conduct a few qualitative experiments on the CelebA dataset: (i) Generation by transitions in the latent space along the direction of a particular attribute, (ii) transitions in the latent vectors between two generated samples and (iii) plot of the Nearest neighbour samples for a given generated image, from the training set. Note that all the interpolations are done in the latent space.

**Results:** The outcome of these experiments are shown in in Figure 2a, 2b, 2c and 2d. Each row in (a) and (b) presents manipulation of a particular face attribute (Big Nose, Heavy Makeup, Black Hair, Smiling, Male). The middle image in each row of (a) corresponds to a training sample with the attribute present and the middle image of a row in (b) represents a sample without the attribute. Each row in (c) represents linear interpolation in the latent space between two randomly selected test samples in the first and the last column. The interpolation results presented in (a), (b), and (c) clearly depicts the smoothness of the learnt latent space of FlexAE as it provides provides smooth transition between any two random images. The first image in each row in (d) shows a randomly generated sample using FlexAE and the next four entries are the four nearest neighbours from the training split. Visual dissimilarity between any generated image and its nearest neighbours from the training split confirms that FlexAE has not merely memorized the training set. (cf. Sec. 10 of the supplementary for more qualitative results).

Figure 2: Interpolations in the latent space of FLexAE on CelebA. Each row in (a) and (b) presents manipulation of a particular face attribute (Big Nose, Heavy Makeup, Black Hair, Smiling, Male). The middle image of each row of (a) and (b) is a true image from the train and test split with and without the attribute respectively. Each row in (c) represents linear interpolation in the latent space between two randomly selected test samples in the first and the last entry. The first image in each row in (d) shows a randomly generated sample using FlexAE and the next four entries are the four nearest neighbours from the training split.

5 Conclusion

In this paper, we systematically studied the effect of the latent prior on the AE-based generative models. We demonstrated that fixing any kind of prior in a data-agnostic way is detrimental to the performance. We also showed that with finite data, there exists a bias-variance trade-off with imposition of any prior on the latent space. We proposed a model called the FlexAE that can potentially operate at different points of the bias-variance curve, and empirically demonstrated its efficacy. We believe that an interesting future direction is to explore latent identifiability using similar principles.
Supplementary Material

6 Theoretical Results

6.1 Breakup of LLE

\[ LLE(\theta) = \mathbb{E}_{p_d(x)} \left[ \log p_\theta(x) \right] - \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x) \right] = \mathbb{E}_{p_d(x)} \left[ \log p_\theta(x, z) \right] - \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x, z) \right] \]

\[ = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right] + \mathbb{E}_{p_d(x)} \left[ D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \right] \]

\[ \overset{(a)}{=} \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log \frac{p_\theta(x|z)p_\phi(z)}{q_\phi(z|x)q_\phi(z)} \right] + \mathbb{E}_{p_d(x)} \left[ D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \right] \]

\[ = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - D_{KL}(q_\phi(z) || p_\phi(z)) - I(x; z_\phi) + \mathbb{E}_{p_d(x)} \left[ D_{KL}(q_\phi(z|x) || p_\theta(z|x)) \right] \]

6.2 Theorems and Proof:

**Theorem 1.** Let the optimal parameters which maximize the ELBO term be \((\phi^*, \theta^*, \psi^*) = \arg \max_{\phi, \theta, \psi} ELBO(\theta, \phi, \psi)\). Then the following necessary and sufficient conditions hold:

\begin{align*}
C1: & \quad p_{\phi^*}(z) = q_{\phi^*}(z) \\
C2: & \quad p_d(x)q_{\phi^*}(z|x) = q_{\phi^*}(z)p_\theta(x|z) \\
C3: & \quad ELBO(\phi^*, \theta^*, \psi^*) = -H(x) \\
C4: & \quad \Delta(\theta^*, \phi^*) = 0
\end{align*}

**Proof:**

Consider the following chain of equations and inequalities:

\[ \begin{align*}
ELBO(\theta, \phi, \psi) & = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - I(x; z_\phi) - D_{KL}(q_\phi(z) || p_\phi(z)) \\
& = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log \frac{q_\phi(z|x)}{q_\phi(z)} \right] - D_{KL}(q_\phi(z) || p_\phi(z)) \\
& = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log \frac{p_\theta(x|z)q_\phi(z)}{q_\phi(z|x)q_\phi(z)} \right] - D_{KL}(q_\phi(z) || p_\phi(z)) \\
& = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_d(x) \right] + \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log \frac{p_\theta(x|z)q_\phi(z)}{q_\phi(z|x)p_d(x)} \right] - D_{KL}(q_\phi(z) || p_\phi(z)) \\
& = -H(x) - D_{KL}(p_d(x)q_\phi(z|x)||q_\phi(z)p_\theta(x|z)) - D_{KL}(q_\phi(z)||p_\phi(z)) \\
& \overset{(a)}{=} -H(x)
\end{align*} \]

\[ \leq -H(x), \]
We first prove Theorem 2 via contradiction. Suppose the auto-encoder with the above choice of Decoder, $p_{\psi}(z) = q_{\phi}(z)$ and $p_{d}(x)q_{\phi}(z|x) = q_{\phi}(z)p_{0}(x|z)$. Let us assume $\phi^*, \theta^*$, and $\phi^*$ denote the optimal parameters for which $p_{\psi^*}(z) = q_{\phi^*}(z), p_{d}(x)q_{\phi^*}(z|x) = q_{\phi^*}(z)p_{0^*}(x|z)$ and ELBO achieves its maximum value $\text{ELBO}(\theta^*, \phi^*, \psi^*) = -H(x)$. This proves the first three conditions: C1, C2, and C3 of Theorem 1.

Theorem 3. If true data distribution $p_d(x)$ is not Gaussian, then under the assumption of Gaussian Decoder, $p_{0}(x|z) \sim \mathcal{N}(\mu, \Sigma)$ and Gaussian prior, $q_{\psi_0}(z) \sim \mathcal{N}(0, I)$, an AE-based model cannot reach the optimum value in ELBO maximization and hence cannot maximize the likelihood.

Proof: We first prove Theorem 2 via contradiction. Suppose the auto-encoder with the above choice of Gaussian prior and Gaussian decoder achieves the optimum value of the ELBO. Let the optimum parameters of such an auto-encoder be $(\hat{\theta}, \hat{\phi})$. Thus as per the necessary and sufficient condition of Theorem 1, the aggregated posterior of the encoder, $q_{\hat{\phi}}(z)$ matches the Gaussian prior, $q_{\psi_0}(z)$ perfectly, and $p_{d}(x)q_{\hat{\phi}}(z|x) = q_{\hat{\phi}}(z)p_{0}(x|z)$.

Since now $q_{\hat{\phi}}(z)$ is Gaussian and decoder is also gaussian, the RHS of above equation represents a jointly gaussian distribution.

From the closure property of multivariate Gaussian distribution, marginal and conditional distribution of a jointly Gaussian distribution are always Gaussian (refer to section 6.2.1) and can be uniquely defined in terms of the mean and variance of the joint distribution. This would imply the true data distribution, $p_d(x)$ must be Gaussian, which contradicts the initial assumption. Proof of Theorem 3 follows a similar argument as that of Theorem 2. $\square$
6.2.1 Marginal and conditional of a joint Gaussian is Gaussian

Suppose that,
\[
\begin{bmatrix} x \\ z \end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix} \mu_x \\ \mu_z \end{bmatrix}, \begin{bmatrix} \Sigma_{xx} & \Sigma_{xz} \\ \Sigma_{zx} & \Sigma_{zz} \end{bmatrix} \right)
\] (10)

Where, \( x \in \mathbb{R}^d \) and \( z \in \mathbb{R}^m \). Then, the marginal and the conditional densities are also Gaussian and given as:
\[
x \sim \mathcal{N}(\mu_x, \Sigma_{xx})
\] (11)
\[
z \sim \mathcal{N}(\mu_z, \Sigma_{zz})
\] (12)
\[
x \mid z \sim \mathcal{N}\left( \mu_x + \Sigma_{xz} \Sigma_{zz}^{-1} (z - \mu_z), \Sigma_{xx} - \Sigma_{xz} \Sigma_{zz}^{-1} \Sigma_{zx} \right)
\] (13)
\[
z \mid x \sim \mathcal{N}\left( \mu_z + \Sigma_{zx} \Sigma_{xx}^{-1} (x - \mu_x), \Sigma_{zz} - \Sigma_{zx} \Sigma_{xx}^{-1} \Sigma_{xz} \right)
\] (14)

**Theorem 4.1.** With sufficient capacity Encoders and Decoders, the AE-model without any prior or regularization on the latent space will achieve the optimal likelihood.

**Proof:**
Since there is no prior imposition in this architecture, one can split the log-likelihood term as the following combination:
\[
LLE(\theta) = \underbrace{\mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right]}_{ELBO(\theta, \phi)} - I(x; z_\phi) + \underbrace{\mathbb{E}_{p_d(x)} \left[ D_{KL}(q_\phi(z|x)||p_\theta(z|x)) \right]}_{\Delta(\theta, \phi)}
\] (15)

Thus, in contrast to our model with allowance of a general prior, the decomposition here involves ELBO term which is only the function of \((\theta, \phi)\), while the second \(\Delta(\theta, \phi)\) remains of the same form.

The ELBO term can be further simplified:
\[
ELBO(\theta, \phi) = -H(x) - D_{KL}(p_d(x)q_\phi(z|x)||q_\phi(z)p_\theta(x|z)),
\] (16)

which is maximized when \(p_d(x)q_\phi(z|x) = q_\phi(z)p_\theta(x|z)\), where \(q_\phi(z) = \int_x p_d(x)q_\phi(z|x)\). In other words, ELBO is maximized (and as we saw earlier, LLE will be maximized) if for every encoder parameter, \(\phi\), a decoder parameter \(\theta\) can be found such that:
\[
p_\theta(x|z) = \frac{p_d(x)q_\phi(z|x)}{\int_x p_d(x)q_\phi(z|x)},
\] (17)

which is possible if we allow for decoders with large enough capacity.

6.3 Joint Variational Encoding and Adversarial Generation

One can consider a general form of the ELBO function as above in the following way to allow for variants for general distance metrics as a result of optimization via generator-critic pair as in the Figure[3]

\[
\mathcal{L}(\theta, \phi, \psi, \kappa) = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - \underbrace{V(\psi; \kappa)}_{\text{GAN Loss}} - I(x; z_\phi),
\] (18)
where \( \psi \) and \( \kappa \) are generator and critic parameters respectively. Thus the joint optimization problem, is the max-min optimization:

\[
\max_{\phi, \theta, \psi} \min_{\theta, \phi, \psi, \kappa} \mathcal{L}(\theta, \phi, \psi, \kappa) = \max_{\theta, \phi, \psi} \left( \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - \min_{\kappa} \mathbb{E}_{\kappa} V(\psi, \kappa) - I(x; z_\phi) \right) - I(x; z_\phi)
\]

(19)

For the vanilla GAN training as in [38], the Discriminator (Critic) Loss (ignoring the additive constants) is proportional to the Jensen-Shannon Divergence between the generated prior and the encoded prior, more precisely it is \( 2D_JS(q_\phi(z)||p_\psi(z)) \). In case the objective is that of Wasserstein metric as in [27], the Critic Loss, instead is the Wasserstein loss between the two priors, \( D_W(q_\phi(z)||p_\psi(z)) \). Thus with some abuse of terminology, we can define the following equivalent terms:

\[
ELBO_{GAN}(\theta, \phi, \psi) = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - 2D_JS(q_\phi(z)||p_\psi(z)) - I(x; z_\phi)
\]

(20)

\[
ELBO_{FlexAE}(\theta, \phi, \psi) = \mathbb{E}_{p_d(x)q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] - D_W(q_\phi(z)||p_\psi(z)) - I(x; z_\phi)
\]

(21)

Due to practical considerations, instead of ELBO maximization, we resort to maximization of the objective in (21). The theorem below justify its use:

**Theorem 5.** For the general setup described in Fig. [3] let the optimal parameters which maximize the \( ELBO_{GAN} \) and \( ELBO_{WGAN} \) term be \((\phi^*_G, \theta^*_G, \psi^*_G) = \arg \max_{\phi, \theta, \psi} ELBO_{GAN}(\theta, \phi, \psi) \) and \((\phi^*_W, \theta^*_W, \psi^*_W) = \arg \max_{\phi, \theta, \psi} ELBO_{WGAN}(\theta, \phi, \psi) \), respectively. Then both the tuples, \((\phi^*_G, \theta^*_G, \psi^*_G)\) and \((\phi^*_W, \theta^*_W, \psi^*_W)\) satisfy the necessary and sufficient conditions mentioned in the Theorem [7].

**Proof:**
Proof follows from similar arguments in the proof of Theorem [1] and noting that both \( D_JS(\cdot||\cdot) \) and \( D_W(\cdot||\cdot) \) are distance metrics.

### 6.4 Loss Functions of FlexAE

In this section, we present the loss functions corresponding to the terms a, b, and c in Equation (21). During training the proposed generative framework FlexAE, we minimize these surrogate loss functions to maximize the \( ELBO_{FlexAE} \) as in Equation (21).

1. **Likelihood Loss (Term a):**

\[
L_{AE} = \frac{1}{s} \sum_{i=1}^s \|x^{(i)} - D_\psi(E_\phi(x^{(i)}))\|
\]

(22)

2. **Wasserstein Loss (Term b) [27]:**

\[
L_{Critic} = \frac{1}{s} \sum_{i=1}^s C_\kappa(\hat{z}^{(i)}) - \frac{1}{s} \sum_{i=1}^s C_\kappa(z^{(i)}) + \frac{\beta}{s} \sum_{i=1}^s (\|\nabla_{z^{(i)}} C_\kappa(z^{(i)})\| - 1)^2
\]

(23)

\[
L_{Gen} = -\frac{1}{s} \sum_{i=1}^s C_\kappa(\hat{z}^{(i)})
\]

(24)

\[
L_{Enc} = \frac{1}{s} \sum_{i=1}^s C_\kappa(z^{(i)})
\]

(25)

12
3. MI Loss (Term c) \[28\]:

\[
L_{MI} = \mathbb{E}_{q_\theta(x,z)} \text{softplus}( - R_\phi(x^{(i)}, z^{(i)})) + \mathbb{E}_{p_\phi(x|z)} \text{softplus}(R_\phi(x^{(i)}, z^{(i)})) \tag{26}
\]

\[
I(X;Y) = \mathbb{E}_{q_\theta(x,z)} R_\phi(x^{(i)}, z^{(i)}) - \log \left( \mathbb{E}_{p_\phi(x|z)} \text{clip}(e^{R_\phi(x^{(i)}, z^{(i)}), e^{-\tau}, e^\tau}) \right) \tag{27}
\]

As in many previous works \[1, 2, 5, 6\], we have assumed a Gaussian decoder, \(D_\theta\), we use mean squared error (Equation 21) as a surrogate loss function for the likelihood loss (term a) in Equation 1. In Equation 22, \(x^{(i)}\) and \(s\) respectively denote the \(i^{th}\) training sample and the number of training points.

In the distribution matching loss or Wasserstein loss \[27, 39\] objectives corresponding to term b in Equation 21 (Equation 23 \[24\] and \[25\], \(C_n\) denotes the critic network. As mentioned in \[27\], the critic network, \(C_n\) in a WGAN must lie withing the space of 1-Lipschitz function. In order to enforce the Lipschitz constraint on the critic network, we implement gradient penalty (third term in Equation 23) as in \[39\]. \(z^{(i)} = E_\phi(x^{(i)}), \tilde{z}^{(i)} = G_\phi(n^{(i)})\) and \(n^{(i)} \sim \mathcal{N}(0, I)\), \(z^{(i)}_\text{avg} = \alpha z^{(i)} + (1 - \alpha)\tilde{z}^{(i)}\), \(\alpha, \beta\) are hyper parameters, with \(\alpha \sim U[0, 1]\), and \(\beta\) as in \[39\].

For the MI loss (term c in Equation 21), two sampling procedures are used - \(x^{(i)}, z^{(i)} \sim q_\phi(x, z)\), and \(x^{(i)}_k, z^{(i)}_k \sim p_\phi(x_k|z_k)\). A lower bound on Jensen-Shannon divergence as in \[40, 28\] is used as a loss \(L_{MI}\) to obtain density ratios while optimizing the regression network, \(R_\phi\). The final estimation of MI uses Donsker-Varadhan bound \[40\], with clipping regularizer on the estimator \[28\].

7 Details of Datasets

7.1 Synthetic Dataset

Synthetic data has been generated using a two step process. The steps involved in creating the first dataset where the true latent space is GMM are listed below.

1. Step 1: Six two-dimensional Gaussian distributions are used to generate true latent space of the synthetic dataset. \(z_{11}^{(i)}\) and \(z_{12}^{(i)}\) denotes the \(1^{st}\) and the \(2^{nd}\) dimensions of the \(i^{th}\) sample from the \(k^{th}\) distribution respectively. The distributions are as mentioned below:

\[
\begin{bmatrix}
  z_{11}^{(i)} \\
  z_{12}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right); \\
\begin{bmatrix}
  z_{21}^{(i)} \\
  z_{22}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right);
\]

\[
\begin{bmatrix}
  z_{31}^{(i)} \\
  z_{32}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -5 \\ 5 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\right); \\
\begin{bmatrix}
  z_{41}^{(i)} \\
  z_{42}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ -1.5 \end{bmatrix}, \begin{bmatrix} 0.95 & 0 \\ 0 & 0.95 \end{bmatrix}\right);
\]

\[
\begin{bmatrix}
  z_{51}^{(i)} \\
  z_{52}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -2 \\ -7 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\right); \\
\begin{bmatrix}
  z_{61}^{(i)} \\
  z_{62}^{(i)}
\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} -4 \\ -3 \end{bmatrix}, \begin{bmatrix} 0.75 & 0 \\ 0 & 0.75 \end{bmatrix}\right).
\]

2. Step 2: Next, a three layer MLP is used to map the two-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 \text{tanh}, \text{exp}, \text{tanh} respectively. Weight and bias parameters of each layer is drawn randomly from the following three distributions respectively: \(\mathcal{N}(0, 0.05)\), \(\mathcal{N}(0, 0.2)\), \(\mathcal{N}(0, 0.1)\).

For creating the second dataset, we have used a single Gaussian distribution with \(\mu = 5\), and \(\sigma = 1.5\) in step 1. In step 2, the MLP contains only one linear layer, with weight and bias parameters sampled from \(\mathcal{N}(0, 0.05)\). Because there is no non-linearity involved, the generated data points belong to some Gaussian distribution as under affine transformation one Gaussian distribution is mapped to another Gaussian distribution.
7.2 Real Dataset

The MNIST [32] database of gray scale handwritten digits consists of 60000 training examples and 10000 test samples. The CIFAR-10 [33] 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) [34] 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 [22, 8, 6, 5] in generative model. Table 4 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.

| Dataset     | Dimension (h × w × c) | Train Split Size | Test Split Size |
|-------------|-----------------------|------------------|-----------------|
| MNIST [32]  | 28 × 28 × 1           | 60000            | 10000           |
| CIFAR-10 [33]| 32 × 32 × 3           | 50000            | 10000           |
| CELEBA [34] | 64 × 64 × 3           | 162770           | 19962           |

8 Network Architectures

Figure 3 illustrates the components present in our proposed FlexAE generative framework. Like any other AE based generative model, it has a reconstruction pipeline consisting of an encoder ($E_\phi$) and a decoder ($D_\theta$) network. We have introduced a generative adversarial network consisting of a generator network ($G_\psi$) and a critic network ($C_\kappa$) to facilitate sampling from the latent space of the reconstruction pipeline. The regression network, $R_\zeta$ is used to estimate the mutual information between an input to the reconstruction pipeline and its intermediate latent representation. The generation pipeline involves the latent generator, $G_\psi$ and the image generator, $D_\theta$, meaning generation is a two-step process. First, we sample from the latent space using the latent generator, $G_\psi$. Next, the image generator, $D_\theta$ samples from the image space using the generated latent code.

Next, we describe the architectures of each of the components in Figure 3 used for the synthetic and the real experiments.

8.1 Synthetic Experiment

Table 5 presents architectures of different networks used in conducting the synthetic experiment. VAE [1] consists of only encoder and decoder. WAE [4] consists of encoder, decoder and critic. FlexAE involves all the networks.

| Encoder | Decoder | Generator | Critic | MI-Net |
|---------|---------|-----------|--------|--------|
| $x \in \mathbb{R}^{128}$ | $z \in \mathbb{R}^{2}$ | $n \in \mathbb{R}^{2}$ | $z \in \mathbb{R}^{2}$ | $x \in \mathbb{R}^{128}$ | $z \in \mathbb{R}^{2}$ |
| $\rightarrow FC_{128} \rightarrow ReLU$ | $\rightarrow FC_{128} \rightarrow Tanh$ | $\rightarrow FC_{128} \rightarrow ReLU$ | $\rightarrow FC_{128} \rightarrow ReLU$ | $\rightarrow FC_{128} \rightarrow ReLU$ | $\rightarrow FC_4 \rightarrow ReLU$ |
| $\rightarrow FC_2$ | $\rightarrow FC_2$ | $\rightarrow FC_2 \rightarrow ReLU$ | $\rightarrow FC_2 \rightarrow ReLU$ | $\rightarrow FC_4 \rightarrow ReLU$ |

8.2 Real Experiment

For real experiments, the encoder, ($E_\phi$) and the decoder, ($D_\theta$) architectures are adopted from prior work [22]. The architecture of the encoder and the decoder networks vary from one dataset to another as presented in Table 6. However, the architectures of the generator, ($G_\psi$), the critic, ($C_\kappa$) and the regression network or MI-Net, ($R_\zeta$) are fixed across all datasets as mentioned in Table 7. The capacity (no. of trainable parameters) of $G_\psi$ and $C_\kappa$ is fairly small as compared to the AE to ensure that the adversarial training does not overfit the latent space. However, if the capacity of $G_\psi$ and
The number of parameters of the latent generator model is the highest. Therefore, we choose a moderate capacity generator and critic network.

Table 8 lists the architectures of different capacity generators used in the bias-variance experiment (Sec. 4.2 in the main paper). Please note that the number of parameters of the latent generator model increases with model number in Table 8. Thus, the capacity of the Model-1 is the least and the capacity of the Model-6 is the highest.

| Encoder | Decoder |
|---------|---------|

**Table 6: Encoder and Decoder Architectures for Real Datasets**

**MNIST**  | **CIFAR10**  | **CELEBA**  |
---|---|---|
**MNIST**  | **CIFAR10**  | **CELEBA**  |

Figure 3: The auto-encoder block of FlexAE projects data into a low dimensional space and reconstruct. The GAN introduces an additional state-space model in the latent space to regularize the latent space of the auto-encoder and learn the prior flexibly.
Table 7: Generator, Critic and MI-Net Architectures for Real Datasets

| Generator | Critic | MI-Net |
|-----------|--------|--------|
| \( n \in \mathbb{R}^m \) → FC\(_{1024}\) → ReLU | \( z \in \mathbb{R}^m \) → FC\(_{512}\) → ReLU | \( x \in \mathbb{R}^{h \times w \times c} \) → Conv\(_{64,3,2}\) → ReLU → FC\(_{512}\) → ReLU |
| \( m \) → FC\(_{512}\) → ReLU | \( m \) → FC\(_{256}\) → ReLU | \( m \) → FC\(_{128}\) → ReLU → FC\(_{1024}\) → ReLU |
| \( m \) → FC\(_{m}\) | \( m \) → FC\(_{128}\) → ReLU | \( m \) → FC\(_m\) \→ FC\(_{512}\) → ReLU → FC\(_{512}\) → ReLU |
| \( m \) → FC\(_1\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_1\) |

\( m = 32 \) for MNIST and \( m = 128 \) for CIFAR10, CELEBA.

Table 8: Generator Architectures for Bias-Variance Experiment

| Model 1 | Model 2 | Model 3 | Model 4 | Model 5 | Model 6 |
|---------|---------|---------|---------|---------|---------|
| \( n \in \mathbb{R}^m \) → FC\(_{16}\) → ReLU | \( n \in \mathbb{R}^m \) → FC\(_{64}\) → ReLU | \( n \in \mathbb{R}^m \) → FC\(_{128}\) → ReLU | \( n \in \mathbb{R}^m \) → FC\(_{256}\) → ReLU | \( n \in \mathbb{R}^m \) → FC\(_{512}\) → ReLU | \( n \in \mathbb{R}^m \) → FC\(_{512}\) → ReLU |
| \( m \) → FC\(_{16}\) → FC\(_{64}\) → ReLU | \( m \) → FC\(_{128}\) → FC\(_{256}\) → ReLU | \( m \) → FC\(_{256}\) → FC\(_{512}\) → ReLU | \( m \) → FC\(_{256}\) → FC\(_{512}\) → ReLU | \( m \) → FC\(_{256}\) → FC\(_{512}\) → ReLU | \( m \) → FC\(_{256}\) → FC\(_{512}\) → ReLU |
| \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) |
| \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) | \( m \) → FC\(_{m}\) |

9 Training Algorithm, Hyper-parameters, Computing Resource and Average Runtime

As mentioned in the main paper, the auto-encoder is required to be optimized jointly with the GAN 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 five losses specified in the main paper in every training iteration. Specifically, in each learning loop, we optimize the \( L_{AE} \), \( L_{Critic} \), \( L_{Gen} \), \( L_{MI} \), 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 Zotac GeForce\(^\circledR\) GTX 1080 Ti 11GB Graphic Card for all of our experiments. The average runtime for experiments on MNIST, CIFAR-10, and CELEBA is approximately 20 hours, 40 hours and 86 hours respectively.

10 Experimental Results

In the main paper, the performance of FlexAE is evaluated mainly quantitatively, using standard metrics: FID [25] and precision/recall [37] 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 FlexAE outperforms all other current state-of-the-art AE based generative models as measured using those metrics. In this section, we present more qualitative results (reconstruction on test examples, generated samples and resulting images due to interpolation in the latent space) for visual evaluation of the proposed generative framework, FlexAE.

Figure 4a represents reconstruction of 6 randomly chosen samples from test test split of MNIST (row 1 and 2), CIFAR-10 (row 3 and 4), and CELEBA (row 5 and 6) dataset. The odd rows represent true data and the even rows represents reconstructed data. Figure 4b, 4c, 4d represents 36 randomly generated samples of MNIST, CIFAR-10 and CELEBA datasets respectively.
Algorithm 1 Pseudo code for the training loop of FlexAE

Hyper-parameters: \( \eta_{AE} = 0.001, \eta_{Critic} = 0.0001, \eta_{Gen} = 0.0005, \eta_{MI} = 0.0001, \eta_{Enc} = 0.00001, \text{AE\_OPT} = \text{Adam}(lr = \eta_{AE}, \beta_1 = 0.9, \beta_2 = 0.999), \text{CRITIC\_OPT} = \text{Adam}(lr = \eta_{Critic}, \beta_1 = 0.9, \beta_2 = 0.9), \text{GEN\_OPT} = \text{Adam}(lr = \eta_{Gen}, \beta_1 = 0.9, \beta_2 = 0.9), \text{MI\_OPT} = \text{Adam}(lr = \eta_{MI}, \beta_1 = 0.9, \beta_2 = 0.999), \text{ENC\_OPT} = \text{Adam}(lr = \eta_{Enc}, \beta_1 = 0.9, \beta_2 = 0.9), \text{disc\_training\_ratio} = 5. \)

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: end for
7: Minimize \( L_{Gen} \) and Update \( \psi \)
8: Minimize \( L_{MI} \) and Update \( \zeta \)
9: Minimize \( L_{Enc} - L_{MI} \) and Update \( \phi \)
10: end for
11: end function

Next, we present more attribute based interpolation results from the CELEBA test split in Figure 5, Figure 6, Figure 7, Figure 8, and Figure 9 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 a negative test example from the CELEBA dataset i.e. a test sample without the corresponding attribute. Whereas, the central image in the grid of the sub-figures (c) and (d) presents a positive test example i.e. a test sample with the particular attribute. For latent space traversal along a particular attribute direction, we calculate the average representation \( (z_{pos}) \) code 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 the face attributes without supervision. The training was completely unsupervised without using any label information. As can be seen from the Figures 5 - Figures 9, 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 10 presents a 15 \times 15 grid, where, the first column plots some randomly generated face images and the remaining entries in each row are the 14 nearest neighbours (in terms of Euclidean distance) from the training split. 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 4: (a) Visualization of reconstruction quality of FlexAE model on randomly selected data from the test split of MNIST (first and second rows), CIFAR-10 (third and fourth rows) and CELEBA (fifth and sixth rows). The odd rows represent the real data and the even rows represent reconstructed data. Randomly generated samples from (b) MNIST, (c) CIFAR-10, and (d) CELEBA datasets using FlexAE model.

Figure 5: Interpolations in the latent space of FlexAE 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 from the test split without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image from the test split with the attribute.

Figure 6: Interpolations in the latent space of FlexAE 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 from the test split without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image from the test split with the attribute.
Figure 7: Interpolations in the latent space of FlexAE 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 from the test split without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image from the test split with the attribute.

Figure 8: Interpolations in the latent space of FlexAE 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 from the test split without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image from the test split with the attribute.

Figure 9: Interpolations in the latent space of FlexAE 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 from the test split without the attribute. Whereas, the central image of each grid in (c) and (d) is a true image from the test split with the attribute.
Figure 10: The first entry in each row represents a randomly generated face using FlexAE. The remaining entries in each row represent 14 nearest neighbours (in terms of Euclidean distance) from the train split of CELEBA dataset. 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.
References

[1] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[2] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, “β-VAE: Learning basic visual concepts with a constrained variational framework,” in *Proc. of ICLR*, 2017.

[3] A. Makhzani, J. Shlens, N. Jaitly, and I. Goodfellow, “Adversarial autoencoders,” in *Proc. of ICLR*, 2016.

[4] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Scholkopf, “Wasserstein auto-encoders,” in *Proc. of ICLR*, 2018.

[5] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. Black, and B. Scholkopf, “From variational to deterministic autoencoders,” in *Proc. of ICLR*, 2020.

[6] B. Dai and D. Wipf, “Diagnosing and enhancing vae models,” in *Proc. of ICLR*, 2019.

[7] A. Klushyn, N. Chen, R. Kurle, B. Cseke, and P. van der Smagt, “Learning hierarchical priors in VAEs,” in *Proc. of NeurIPS Workshop*, 2019.

[8] A. K. Mondal, S. P. Chowdhury, A. Jayendra, P. Singla, H. Asnani, and A. Prathosh, “MaskAAE: Latent space optimization for adversarial auto-encoders,” in *Proc. of UAI*, 2020.

[9] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, “Autoencoding beyond pixels using a learned similarity metric,” in *Proc. of ICML*, 2016.

[10] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, “Understanding disentangling in β-VAE,” in *NeurIPS Workshop*, 2017.

[11] J. S. Shengjia Zhao and S. Ermon, “Towards deeper understanding of variational autoencoding models,” *arXiv preprint arXiv:1702.08658*, 2017.

[12] S. Zhao, J. Song, and S. Ermon, “InfoVAE: Balancing learning and inference in variational autoencoders,” in *Proc. of AAAI*, 2019.

[13] D. J. Rezende and F. Viola, “Taming vaes,” *arXiv preprint arXiv:1810.00597*, 2018.

[14] J. M. Tomczak and M. Welling, “VAE with a vampprior,” in *Proc. of AISTATS*, 2018.

[15] H. Kim and A. Mnih, “Disentangling by factorising,” in *Proc. of ICML*, 2018.

[16] E. Nalisnick, L. Hertel, and P. Smyth, “Approximate inference for deep latent gaussian mixtures,” in *NeurIPS Workshop*, 2016.

[17] H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi, “Variational autoencoder with implicit optimal priors,” in *Proc. of AAAI*, 2019.

[18] M. Rosca, B. Lakshminarayanan, and S. Mohamed, “Distribution matching in variational inference,” *arXiv preprint arXiv:1802.06847*, 2018.

[19] E. Nalisnick and P. Smyth, “Stick-breaking variational autoencoders,” in *Proc. of ICLR*, 2017.

[20] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, “Improved variational inference with inverse autoregressive flow,” in *Proc. of NeurIPS*, 2016.

[21] D. J. Rezende and S. Mohamed, “Variational inference with normalizing flows,” in *Proc. of ICML*, 2015.

[22] A. Kumar, B. Poole, and K. Murphy, “Regularized autoencoders via relaxed injective probability flow,” *arXiv preprint arXiv:2002.08927*, 2020.

[23] M. Bauer and A. Mnih, “Resampled priors for variational autoencoders,” in *Proc. of AISTATS*, 2019.

[24] C.-W. Huang, A. Touati, L. Dinh, M. Drozdzal, M. Havaei, L. Charlin, and A. Courville, “Learnable explicit density for continuous latent space and variational inference,” *arXiv preprint arXiv:1710.02248*, 2017.

[25] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” in *Proc. of NeurIPS*, 2017.

[26] A. R. Sanjeev Arora and Y. Zhang, “Do gans actually learn the distribution? an empirical study,” in *Proc. of ICLR*, 2018.
[27] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proc. of ICML*, pp. 214–223, 2017.

[28] J. Song and S. Ermon, “Understanding the limitations of variational mutual information estimators,” in *Proc. of ICLR*, 2020.

[29] A. K. Mondal, A. Bhattacharya, S. Mukherjee, S. Kannan, H. Asnani, and Prathosh, “C-mi-gan: Estimation of conditional mutual information using minmax formulation,” *arXiv preprint arXiv:2005.08226*, 2020.

[30] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, “Mine: mutual information neural estimation,” *arXiv preprint arXiv:1801.04062*, 2018.

[31] M. Arjovsky and L. Bottou, “Towards principled methods for training generative adversarial networks,” in *Proc. of ICLR*, 2017.

[32] Y. Lecun, “The mnist database of handwritten digits.” [http://yann.lecun.com/exdb/mnist/](http://yann.lecun.com/exdb/mnist/), 2010.

[33] A. Krizhevsky, “Learning multiple layers of features from tiny images,” tech. rep., 2009.

[34] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proc. of ICCV*, 2015.

[35] L. Theis, A. v. d. Oord, and M. Bethge, “A note on the evaluation of generative models,” in *Proc. of ICLR*, 2016.

[36] M. Lucic, K. Kurach, M. Michalski, O. Bousquet, and S. Gelly, “Are gans created equal? a large-scale study,” in *Proc. of NeurIPS*, 2018.

[37] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, “Assessing generative models via precision and recall,” in *Proc. of NeurIPS*, 2018.

[38] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Proc. of NeurIPS*, 2014.

[39] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of wasserstein gans,” in *Proc. of NeurIPS*, 2017.

[40] S. Nowozin, B. Cseke, and R. Tomioka, “f-gan: Training generative neural samplers using variational divergence minimization,” in *Proc. of NeurIPS*, 2016.