VAEs meet Diffusion Models: Efficient and High-Fidelity Generation

Kushagra Pandey
Department of Computer Science
Indian Institute of Technology, Kanpur
kushagrap@cse.iitk.ac.in

Avideep Mukherjee
Department of Computer Science
Indian Institute of Technology, Kanpur
avideep@cse.iitk.ac.in

Abhishek Kumar
Google Brain
abhishk@google.com

Piyush Rai
Department of Computer Science
Indian Institute of Technology, Kanpur
piyush@cse.iitk.ac.in

Abstract

Diffusion Probabilistic models have been shown to generate state-of-the-art results on several competitive image synthesis benchmarks but lack a low-dimensional, interpretable latent space, and are slow at generation. On the other hand, Variational Autoencoders (VAEs) have access to a low-dimensional latent space but exhibit poor sample quality. Despite recent advances, VAEs usually require large dimensional hierarchies of the latent codes to generate high-quality samples. We present VAEDM, a novel generative framework for *refining* VAE generated samples using diffusion models while also presenting a novel conditional forward process parameterization for diffusion models. We show that the resulting parameterization can improve upon the unconditional diffusion model in terms of sampling efficiency during inference while also equipping diffusion models with the low-dimensional VAE inferred latent code. Furthermore, we show that the proposed model exhibits out-of-the-box capabilities for downstream tasks like image superresolution and denoising.

1 Introduction

Generative modeling is the task of capturing the underlying data distribution and learning to generate novel samples from a posited explicit/implicit distribution of the data in an unsupervised manner. Variational Autoencoders (VAEs) [17, 30] are a type of explicit-likelihood based generative models which also learn a low-dimensional latent representation for the data. The resulting framework is very flexible and can be used for downstream applications, such as learning disentangled representations [3, 11], semi-supervised learning [18], anomaly detection [28], among others. However, in image synthesis applications, VAE generated samples (or reconstructions) are usually blurry and fail to incorporate high-frequency information [7]. Despite recent advances [4, 29, 38, 40] in improving VAE sample quality, there is still a significant gap in sample quality between VAEs and their implicit-likelihood counterparts like GANs [8, 14, 15, 16].

In contrast, Denoising diffusion probabilistic models (DDPM) [12, 34] have been shown to achieve impressive performance on several image synthesis benchmarks, even surpassing GANs on several such benchmarks [6, 13]. However, conventional diffusion models require an expensive iterative sampling procedure and lack a low-dimensional latent representation, limiting these models' practical applicability for downstream applications.

NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications

In this preliminary work, we present VAEDM, a novel framework which combines the best of both VAEs and DDPMs in an attempt to alleviate the aforementioned issues with both types of model families. We present a *generator-refiner* framework in which blurry samples generated from a VAE are *refined* using a novel conditional DDPM formulation. Using qualitative and quantitative experiments, we show that the proposed method, unlike standard VAEs, can model high frequency image details and is able to generate high-quality samples that can be controlled using the low-dimensional VAE latent space. Moreover, we show that the proposed method requires fewer reverse diffusion sampling steps during inference, and exhibits out-of-the-box generalization to downstream tasks like image super-resolution and denoising.

2 Background

2.1 Variational Autoencoders

VAEs [17] are based on a simple but principled encoder-decoder based formulation which tries to maximize the evidence lower bound (ELBO) of the data log-likelihood, which is intractable to compute in general. The VAE optimization objective can be stated as follows:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathcal{D}_{KL}[q_\phi(z|x) \| p(z)] \tag{1}$$

Under the amortized variational inference scheme, the approximate posterior ($q_\phi(z|x)$) and the likelihood ($p_\theta(x|z)$) distributions can be modeled using deep neural networks with parameters ϕ and θ, respectively using the reparameterization trick [17, 30]. The choice of the prior distribution $p(z)$ is flexible and can vary from a standard Gaussian [17] to more expressive priors like normalizing flows [9, 19, 39].

2.2 Denoising Diffusion Probabilistic Models

DDPMs [12, 34] are latent-variable models consisting of a forward noising process ($q(x_{1:T}|x_0)$) which gradually destroys the structure of the data x_0 and a reverse process (($p(x_{0:T})$)) which learns to recover the original data x_0 from the noisy input. The forward noising process is modeled using a first-order Markov chain with Gaussian transitions and is fixed throughout training, and the noise schedules β_1 to β_T can be fixed or learned. The form of the forward process and some notable properties of the forward process conditional distributions are summarized in the equations below

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{2}$$

The forward process of DDPMs admits a closed form for x_t for any t, as follows

$$q(x_t|x_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \quad \text{where} \quad \alpha_t = (1-\beta_t) \quad \text{and} \quad \bar{\alpha}_t = \prod_t \alpha_t \tag{3}$$

The forward process posteriors are also tractable and are given by

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(\tilde{\mu_t}(x_t, x_0), \tilde{\beta}_t) \tag{4}$$

$$\text{where} \quad \tilde{\mu_t}(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}\beta_t}}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \quad \text{and} \quad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{5}$$

The reverse process can also be parameterized using a first-order Markov chain with a learned Gaussian transition distribution as follows

$$p(x_{0:T}) = p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \quad p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{6}$$

Given a large enough T and a well-behaved variance schedule of β_t, the distribution $q(x_T|x_0)$ will approximate an isotropic Gaussian. We can generate a new sample from the underlying data distribution $q(x_0)$ by sampling a latent from $p(x_T)$ (chosen to be an isotropic Gaussian distribution)

and running the reverse process. As proposed in [12], the reverse process in DDPM is trained to minimize the following upper bound over the negative log-likelihood (See [34] for detailed proofs):

$$\mathbb{E}_q[\mathcal{D}_{KL}(q(x_T|x_0)\|p(x_T)) + \sum_{t>1}\mathcal{D}_{KL}(q(x_{t-1}|x_t,x_0)\|p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)] \quad (7)$$

A notable aspect of the above objective is that all the KL divergences involve Gaussians and, consequently, are available in closed form.

3 Coupling VAE with Diffusion Model

3.1 VAEDM Training Objective

Given a high-resolution image x_0, an auxiliary conditioning signal y to be modelled using a VAE, a latent representation z associated with y, and a sequence of T representations $x_{1:T}$ learned by a diffusion model, the VAEDM generative process can be formulated as follows:

$$p(x_{0:T}, y, z) = p(z)p_\theta(y|z)p_\phi(x_{0:T}|y,z) \quad (8)$$

where θ and ϕ are the parameters of the VAE decoder and the reverse process of the conditional diffusion model, respectively. The log-likelihood of the training data can then be obtained as:

$$\log p(x_0) = \log \int p(x_{0:T}, y, z) dx_{1:T} dy dz \quad (9)$$

Since computation of the likelihood in Eq. (9) is intractable, we can approximate it by computing a lower bound (ELBO) with respect to the joint posterior over the unknowns ($x_{1:T}$, y, z) as:

$$\log p(x_0) \geq \mathbb{E}_{q(x_{1:T},y,z|x_0)}\left[\log \frac{p(x_{0:T},y,z)}{q(x_{1:T},y,z|x_0)}\right] \quad (10)$$

where $q(x_{1:T}, y, z|x_0)$ is the joint posterior over the unknowns. Furthermore, the joint posterior can also be factorized into the following conditional distributions:

$$q(x_{1:T}, y, z|x_0) = q(y|x_0)q_\psi(z|y,x_0)q(x_{1:T}|y,z,x_0) \quad (11)$$

where ψ are the parameters of the VAE recognition network ($q_\psi(z|y,x_0)$). As considered in previous works [12, 34] we keep the DDPM forward process ($q(x_{1:T}|y,z,x_0)$) fixed throughout training. It is trivial to show that plugging the forward process formulation in Eq. (11) yields the following ELBO objective (See Appendix B.1 for proof)

$$\log p(x_0) \geq \mathbb{E}_{q(y|x_0)}[\underbrace{\mathbb{E}_{q_\psi(z|y,x_0)}[p_\theta(y|z)] - \mathcal{D}_{KL}(q_\psi(z|y,x_0)||p(z))}_{\mathcal{L}_{\text{VAE}}}] + \underbrace{\mathbb{H}[q(y|x_0)]}_{\mathcal{H}_{\text{q}}} + \mathbb{E}_{y,z\sim q(y,z|x_0)}\left[\underbrace{\mathbb{E}_{q(x_{1:T}|y,z,x_0)}\left[\frac{p_\phi(x_{0:T}|y,z)}{q_\phi(x_{1:T}|y,z,x_0)}\right]}_{\mathcal{L}_{\text{DDPM}}}\right] \quad (12)$$

Therefore, the VAEDM training objective decomposes into an independent VAE and a conditional DDPM training formulation. It is worth noting that in the above analysis, we have shown y to be inferred from a stochastic process ($q(y|x_0)$) conditioned on the training data, x_0. However, the choice of the conditioning signal y can also be deterministic. We next discuss the choice of the conditioning signal y, the $\mathcal{H}_q$ term, and several parameterization choices for the VAE and the DDPM models.

3.2 Specific Instantiation with a Two-stage Model

Training the joint model in (12) can be challenging. Instead, we make some simplifying design choices: Firstly, we decouple the $\mathcal{L}_{\text{VAE}}$ and $\mathcal{L}_{\text{DDPM}}$ terms, and train them in a sequential two-stage manner. We first train using $\mathcal{L}_{\text{VAE}}$ to optimize the log-likelihood of the conditioning signal y which is taken to be x_0 itself. We then sample a single (fixed) latent code for each training data x_0 using ($\hat{z} \sim q_\psi(z|x_0)$) and then infer the conditioning signal for the next DDPM stage using:

$$\hat{y} = f_\theta(z) \quad (13)$$

where f_θ is the functional form of the VAE decoder, i.e., the mean of the decoder likelihood $p_\theta(x|z)$. More specifically, we use a single sampled z to generate the reconstruction $\hat{y}$ for x_0. Secondly, we condition the next stage DDPM model on $\hat{y}$ instead of y. These design choices ensure a deterministic mapping between x_0 and y, which makes the $\mathcal{H}_{\text{q}}$ term disappear from the training objective, thus simplifying training. Here we only consider a single-stage VAE (with a single stochastic layer) discussed in Section 2.1. However, due to the flexibility of the VAEDM two-stage training, multi-stage VAE approaches as proposed in [4, 29, 38] can also be utilized to model the conditioning signal y.

In the next stage, we then train using $\mathcal{L}_{\text{DDPM}}$ while taking the conditioning signal $\hat{y}$ to be the mean reconstruction from the VAE. More specifically, we use the *simplified* training objective proposed in [12] for training the diffusion model since it empirically leads to better sample quality during inference. We consider the following two variants of conditional DDPM's in this work.

3.2.1 Formulation 1

In this formulation, we make the following simplifying assumptions:

1. We assume that the forward process is independent of the VAE reconstructions y and the latent code information z i.e. $q(x_{1:T}|y, z, x_0) \approx q(x_{1:T}|x_0)$.
2. The reverse process transitions are conditionally dependent on only the VAE reconstruction i.e. $p(x_{0:T}|y, z) \approx p(x_{0:T}|y)$

A similar parameterization has been considered in recent work on conditional DDPM models [13, 32]. We concatenate the VAE reconstruction to the reverse process representation x_t at each time step t to obtain x_{t-1}.

3.2.2 Formulation 2

In this formulation, we make the following simplifying assumptions:

1. The forward process transitions are conditionally dependent on the VAE reconstruction but independent of the VAE latent code information z i.e. $q(x_{1:T}|y, z, x_0) \approx q(x_{1:T}|y, x_0)$
2. The reverse process transitions are conditionally dependent on only the VAE reconstruction i.e. $p(x_{0:T}|y, z) \approx p(x_{0:T}|y)$

More specifically, we design the forward process transitions to incorporate the VAE reconstruction y as follows:

$$q(x_1|x_0, y) = \mathcal{N}(\sqrt{1-\beta_1}x_0 + y, \beta_1 I) \tag{14}$$

$$q(x_t|x_{t-1}, y) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1} + (1-\sqrt{1-\beta_t})y, \beta_t I) \quad \text{where} \quad t > 1 \tag{15}$$

It can be shown that the forward conditional marginal in this case becomes (See Appendix B.2 for proof):

$$q(x_t|x_0, y) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0 + y, (1-\bar{\alpha}_t)I) \tag{16}$$

For $t = T$ and a *well-behaved* noise schedule β_t, $\bar{\alpha}_T \approx 0$ which implies $q(x_T|x_0, y) \approx \mathcal{N}(y, I)$. Intuitively, this means that the Gaussian $\mathcal{N}(y, I)$ becomes our base measure ($p(x_T)$) during inference on which we need to run our reverse process. Since, the simplified denoising training formulation proposed in [12] depends on the functional form of the forward process posterior $q(x_{t-1}|x_t, x_0)$, this formulation results in several modifications in the standard DDPM training and inference which are discussed in detail in Appendix A. We present the effectiveness of the proposed DDPM formulations in subsequent sections.

It is worth noting that in principle, it is also possible to condition the forward and the reverse processes for both the DDPM formulations with the VAE latent code information z. However we do not consider such a conditioning scheme in this work.

4 Comparison with existing approaches

Following the seminal work of [12, 34] in diffusion models, there has been a lot of recent progress in both unconditional [6, 20, 26] and conditional diffusion models [2, 5, 13, 32] (including score-based

Figure 1: Samples generated from VAEDM (Formulations 1 and 2). All samples were generated using $t = 1000$ steps in the reverse process. Bottom row shows the VAE samples corresponding to the generated samples.

models [35, 36], based on a connection proposed in [12]) for a variety of downstream tasks including image synthesis, audio synthesis and likelihood estimation among others. Similarly there has also been progress in improving the ELBO estimates [1, 24, 33] and image synthesis [4, 23, 38, 41] quality using VAE's [17, 30]. Next, we compare our proposed approach in detail with several of these related existing model families.

Unconditional DDPM: DDPM as introduced in [12] generates images unconditionally which has limited application scope. On the other hand, the proposed VAEDM model can be used for tasks like image enhancement, super-resolution etc. Essentially, the DDPM formulations proposed in our method can also be trained in a stand-alone manner with the reverse process conditioned on an auxiliary input like a grayscale image or an image with missing pixels to generate colorized or inpainted results, respectively. Moreover, our approach requires lesser number of reverse process steps in general to produce plausible samples.

Conditional DDPM: Conditional DDPM as introduced in [13] and [32] use multiple diffusion models for generating high-resolution images in a cascaded fashion. However, for even a two-stage pipeline the sampling time of such models would be effectively much higher than VAEDM. Given the flexibility of our approach, we hypothesize that a single-stage VAE can also be replaced by a complex multi-stage VAE architecture as proposed in [4, 38] for comparable sample quality to cascaded diffusion models without affecting the sampling time significantly. Moreover, such cascades lack a low-dimensional latent code which might be a limiting factor for certain downstream applications. It is worth noting that, [13] use a conditioning augmentation scheme where the high resolution image is generated by conditioning on a blurred/noisy low resolution image. This augmentation scheme is empirically chosen and can be sub-optimal across datasets. On the contrary, our model is conditioned on a reconstruction generated by a VAE and no explicit augmentation is required.

Hierarchical VAEs Hierarchical VAEs [4, 29, 37, 38] can suffer from posterior collapse and heuristics like gradient skipping and spectral normalization [25] might be required to stabilize training. Moreover, these models require a large dimensionality of the latent codes to generate high-fidelity samples [29, 40]. In contrast, VAEDM training does not suffer from such instabilities and the proposed method requires a single latent code layer (with dimensionality comparable to GANs) to generate high-fidelity samples. Interestingly, in principle, our approach can be considered similar to Hierarchical VAEs in some sense since the first stage VAE is used to model the high-level attributes of the generated image while the DDPM model in the second stage can be considered as a refiner that adds low-level details. In Hierarchical VAEs, the subsequent latent hierarchies perform this function. [4].

5 Experiments

We evaluate the effectiveness of our method by making qualitative and quantitative comparisons between the proposed VAEDM framework and the unconditional DDPM model [12] on CIFAR10 [21] and the CelebAMask-HQ [22] datasets. All the images in the CelebAMask-HQ dataset were downsampled to 128 x 128 resolution for efficiency. For all the experiments, we set $T = 1000$,

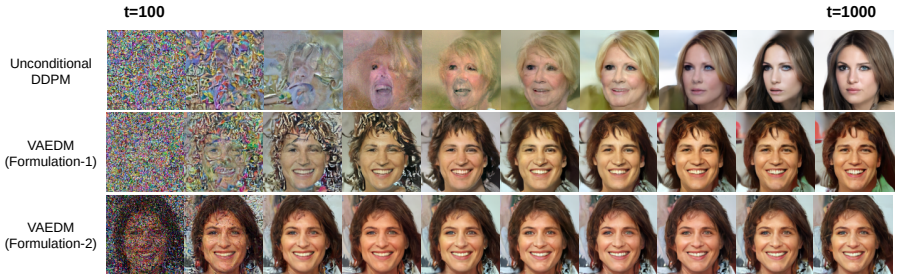

Figure 2: Samples generated from the unconditional DDPM and VAEDM (Formulations 1 and 2). The time t denotes the number of reverse process steps performed during inference. The samples were obtained at equidistant time points between t=100 and t=1000

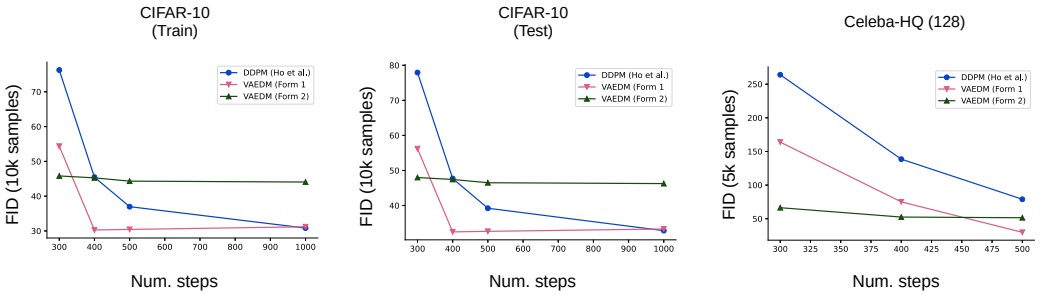

Figure 3: FID score comparison between the unconditional DDPM and VAEDM for CIFAR-10 (train and test) and CelebAMask-HQ (128 x 128). The FID scores were computed using 10k samples for CIFAR-10 and 5k samples for CelebAMask-HQ

$\beta_1 = 10^{-4}$ and $\beta_2 = 0.02$ and use our unconditional DDPM implementation to compare with the proposed approach. For FID score computation, we use the *clean-fid* package as FID scores are sensitive to image resizing operations [27]. More details regarding the network architecture and the training procedure can be found in Appendix C.

5.1 Sample quality

Fig. 1 shows some samples generated from the VAEDM model. As can be observed from the visualization, the final generated samples (Fig. 1 (Top row)) are a refinement of the *blurry* samples generated by our single-stage VAE model (Fig. 1 (Bottom row)). Some more unconditional samples from the VAEDM model are provided in Appendix D.

5.2 Number of Reverse process steps during sampling

Next, we qualitatively compared the unconditional DDPM model proposed in [12] and VAEDM in terms of the number of reverse process sampling steps required to produce high-fidelity samples. Fig. 2 shows the samples generated from the unconditional DDPM (Top) and the samples generated using VAEDM formulations 1 (Middle) and 2 (Bottom). The visualization in Fig. 2 suggests that our method requires considerably fewer number of reverse process steps to generate high-fidelity samples than the unconditional DDPM model. Interestingly, VAEDM with formulation 2 can generate plausible samples starting from $t = 300$. These qualitative results are further supported by Fig. 3 where VAEDM largely outperforms the unconditional DDPM model in terms of sample quality in the low-step regime. For CIFAR-10, VAEDM with formulation-1 matches the FID score of the unconditional DDPM model after only $t = 400$ steps. For a more complex dataset like CelebAMask-HQ, there is an even larger gap in sample quality between the two models. At $t = 300$, the FID score for the unconditional DDPM is around 264.01 and 66.39 for VAEDM with formulation-2. Within the two VAEDM formulations, interestingly, we noticed that the FID scores for formulation-2 remain relatively constant with step-size suggesting it might be more applicable in the low step-size budget

scenario, while formulation-1 can provide the right balance between sample quality and the number of sampling steps required in the reverse process.

Intuitively, these results might not be surprising as the VAE model in VAEDM generates a basic template of the final image (in the form of a blurry reconstruction) which the downstream DDPM model can then refine. On the contrary, the unconditional DDPM requires a longer sampling schedule to model all image details, which is a harder task than refining a pre-generated template. Moreover, since the cost of sampling from a VAE (a single forward pass) is minimal as compared to the downstream DDPM model, we are able to reduce the overall sampling time required to generate plausible samples when compared with the unconditional DDPM model.

5.3 Interpolations in the latent space

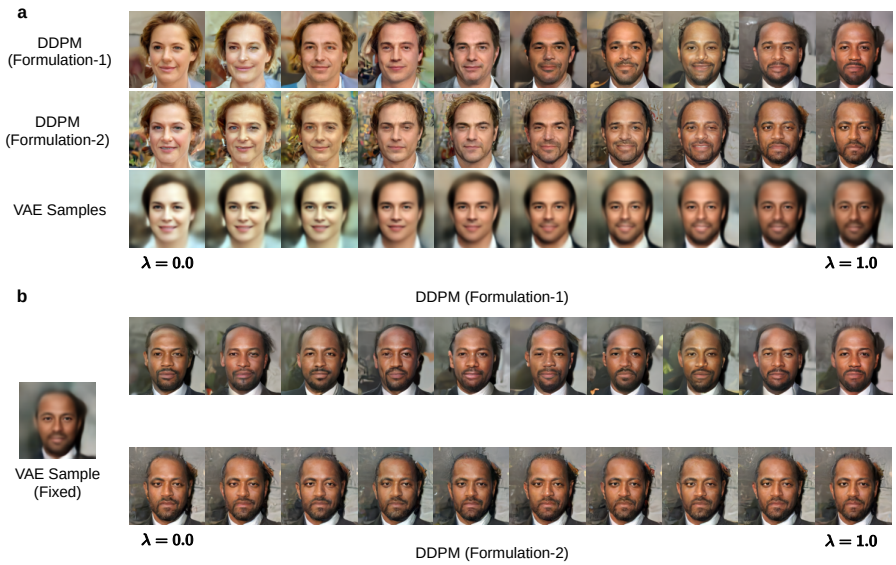

Figure 4: VAEDM interpolation using z_{vae} and z_{ddpm}. Samples generated using t=1000 steps.

The proposed VAEDM model consists of two latent codes: the VAE latent code z_{vae} and the latent representations z_{ddpm} associated with the reverse process base measure $p(x_T)$ (which is of the same size of the input image x_0 and thus might not be beneficial for downstream tasks). We next discuss the effects of manipulating both z_{vae} and z_{ddpm}. We consider the following two interpolation settings:

1. **Varying z_{vae} with fixed z_{ddpm}**: We first sampled two VAE latent codes $z_{vae}^{(1)}$ and $z_{vae}^{(2)}$ using the standard Gaussian distribution. We then performed linear interpolation between $z_{vae}^{(1)}$ and $z_{vae}^{(2)}$ to obtain intermediate VAE latent codes $\tilde{z}_{vae} = \lambda z_{vae}^{(1)} + (1-\lambda) z_{vae}^{(2)}$, which were then used to generate the final VAEDM samples with a shared z_{ddpm} latent.
2. **Fixed z_{vae} with varying z_{ddpm}**: Next, we sampled the VAE latent code z_{vae} using the standard Gaussian distribution. With a fixed z_{vae}, we then sampled two latent DDPM representations $z_{ddpm}^{(1)}$ and $z_{ddpm}^{(2)}$ from the reverse process base measure $p(x_T)$. We then performed linear interpolation between $z_{ddpm}^{(1)}$ and $z_{ddpm}^{(2)}$ with a fixed z_{vae} to generate the final VAEDM samples.

Fig. 4a shows the VAEDM generated samples with a fixed z_{ddpm} and the interpolated z_{vae}. Refinements generated using DDPM formulation-1 (Fig. 4a (Top)) results in abrupt interpolations between the two images (For example, a sudden change in skin tone in the 8th image from the Left.), while the interpolation is a lot smoother using DDPM formulation-2 (4a (Middle)). The VAE generated samples are shown in Fig. 4a (Bottom). More interestingly, with a fixed z_{vae}, we did not notice any perceptual changes in the generated images when using DDPM formulation-2 (Fig. 4b (Bottom)). On the contrary, using DDPM formulation-1 (Fig. 4b (Top)) resulted in changes in more fine-grained

features of the image (like beard style and skin color, among others). This observation hints towards more strong conditioning of the DDPM model on the VAE reconstruction using formulation 2. We next discuss the implications of this observation.

5.4 Equipping DDPM with a low-dimensional latent code

In Section 5.3 we hypothesized that the DDPM conditioning introduced in Section 3.2.2 restricts the DDPM model from generating varying samples with a fixed VAE reconstruction. To further verify our hypothesis, we generated VAEDM samples by fixing z_{vae} and choosing z_{ddpm} with varying random seeds. Fig. 5 shows the corresponding generated samples. Similar to the results observed in Section 5.3, we did not observe any perceptual differences among the samples. This observation implies that we can control the DDPM generated samples primarily by manipulating the low-dimensional VAE latent code z_{vae}. Our work is the first to equip diffusion models with a low-dimensional latent representation to the best of our knowledge.

Figure 5: VAEDM samples generated by fixing z_{vae} and randomly chosen z_{ddpm} (t=1000)

5.5 Out-of-the-box super-resolution and denoising

Recently, [32] showed impressive image super-resolution results using diffusion models. However, such a model is trained explicitly for a single task and may not generalize to other forms of noise in the image. To test if our generator-refiner framework can generalize over different types of noisy inputs, we conditioned the DDPM reverse process on the noisy input (instead of the VAE reconstruction). The samples obtained from such conditioning are visualized in Fig. 6. On a 16 x 16 to 32 x 32 image super-resolution task, our model is able to generate plausible reconstructions (Fig. 6a (Middle row)) when compared to the original CIFAR-10 samples. Similarly, on an image denoising task (with noise std=0.1), our model can denoise the corrupted samples (Fig. 6b (Middle row)). Intuitively, these results can be expected since during training, the proposed VAEDM method learns to refine VAE reconstructions which lack a lot of detail and are inherently blurry. Hence the task of refining these reconstructions might be more challenging than learning to upsample a downsampled version of an image. However, it is worth noting that certain artifacts in the generated reconstructions are evident, leaving scope for improvements.

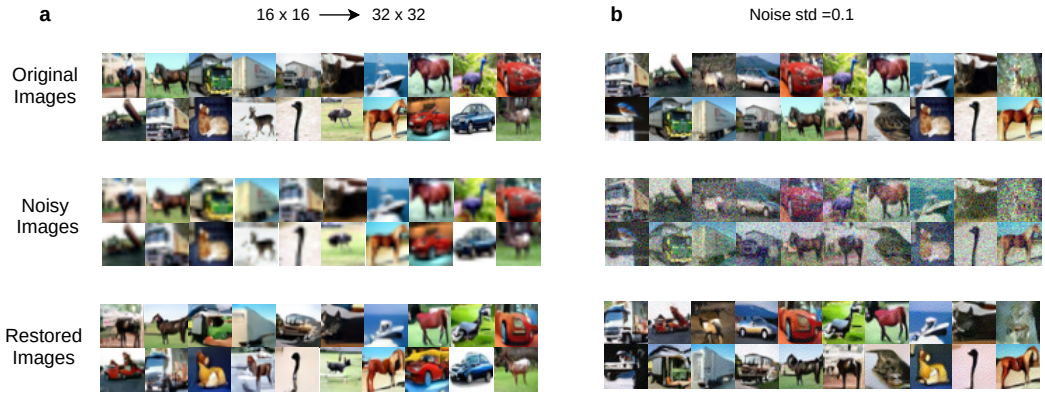

Figure 6: VAEDM (Formulation 2) generated reconstructions for image super-resolution (16x16 to 32x32) (Left) and denoising (t=1000) (Right). The original samples were taken from the CIFAR-10 test set.

6 Conclusion and Future work

In this work, we presented a novel unifying framework for jointly training VAEs and diffusion models. Moreover, we presented two novel conditional parameterizations of diffusion models. We presented the effectiveness of the proposed approach in generating high-quality samples, requiring fewer reverse process steps during inference when compared with the unconditional DDPM formulation, equipping DDPM with a low dimensional latent code, and generalizing to additional tasks like image super-resolution and denoising. However, as pointed in Section 5.5, there is still room for improving VAEDM generalization to auxiliary tasks. Moreover, we also look forward to combining our proposed approach with recent advances in diffusion models and make more quantitative comparisons with other state-of-the-art methods in image synthesis and modeling.

7 Impact statement

We note that the synthetic image generation techniques have the potential to mitigate bias and privacy issues for related ML models that require data collection and annotation. However, such techniques could be misused to produce fake or misleading information, and researchers should be aware of these risks and explore the techniques responsibly.

Acknowledgements

We would like to thank Ben Poole for his insightful comments during the course of this project.

References

[1] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders, 2016.

[2] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation, 2020.

[3] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders, 2019.

[4] Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images, 2021.

[5] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models, 2021.

[6] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[7] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks, 2016.

[8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[9] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models, 2018.

[10] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.

[11] I. Higgins, L. Matthey, A. Pal, Christopher P. Burgess, Xavier Glorot, M. Botvinick, S. Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[13] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021.

[14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2018.

[15] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019.

[16] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan, 2020.

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[18] Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models, 2014.

[19] Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving variational inference with inverse autoregressive flow, 2017.

[20] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models, 2021.

[21] Alex Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[22] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[23] Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation, 2020.

[24] Vaden Masrani, Tuan Anh Le, and Frank Wood. The thermodynamic variational objective, 2021.

[25] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018.

[26] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021.

[27] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On buggy resizing libraries and surprising subtleties in fid calculation. *arXiv preprint arXiv:2104.11222*, 2021.

[28] Adrian Alan Pol, Victor Berger, Gianluca Cerminara, Cecile Germain, and Maurizio Pierini. Anomaly detection with conditional variational autoencoders, 2020.

[29] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2, 2019.

[30] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[32] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021.

[33] Samarth Sinha and Adji B. Dieng. Consistency regularization for variational auto-encoders, 2021.

[34] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics, 2015.

[35] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[36] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

[37] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders, 2016.

[38] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder, 2021.

[39] Rianne van den Berg, Leonard Hasenclever, Jakub M. Tomczak, and Max Welling. Sylvester normalizing flows for variational inference, 2019.

[40] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2018.

[41] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. Vaebm: A symbiosis between variational autoencoders and energy-based models, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 5
   (b) Did you describe the limitations of your work? [Yes] See Section 5.5 and 6
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 7
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sections 3.1,3.2.1,3.2.2
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A,B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code is currently proprietary since the work is on-going
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix C
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Most experiments are qualitative in nature
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix C
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See [21, 22]
   (b) Did you mention the license of the assets? [No] The datasets used are standard benchmarks in the field
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Generated samples from our model
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The datasets used are open-sourced under a license
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The datasets used are standard benchmarks in the field

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix: DDPM training and inference under Formulation 2

Algorithm 1: Training (Form. 2)

repeat
- $x_0 \sim q(x_0)$
- $t \sim$ Uniform($\{1 \dots \mathrm{T}\}$)
- $\epsilon \sim \mathcal{N}(0, I)$
- Take gradient descent step on
  - $\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon + y, t)\|^2$

until *convergence*

Algorithm 2: Inference (Form. 2)

$x_T \sim \mathcal{N}(y, I)$

for ***t = T to 1*** **do**
- $z = \mathcal{N}(0, I),$ if $t > 1$ else 0
- $\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - y - \epsilon_\theta(x_t, y, t)\sqrt{1-\bar{\alpha}_t})$
- $\hat{x}_{t-1} = \gamma_0 \hat{x}_0 + \gamma_1 x_t + \gamma_2 y$
- $x_{t-1} = \hat{x}_{t-1} + z\hat{\sigma}_t$

return $x_0 - y$

The DDPM training objective proposed in [12], has the following form:

$$\mathbb{E}_q[\underbrace{\mathcal{D}_{KL}(q(x_T|x_0)\|p(x_T))}_{L_T} + \sum_{t>1} \underbrace{\mathcal{D}_{KL}(q(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t))}_{L_{t-1}} - \underbrace{\log p_\theta(x_0|x_1)}_{L_0}] \tag{17}$$

Reverse Process parameterization: Following [12], we parameterize the reverse process transition $p_\theta(x_{t-1}|x_t)$ using the functional form of the forward process posterior $q(x_{t-1}|x_t, x_0)$. For the VAEDM formulation proposed in Section 3.2.2, the forward process conditional distributions can be specified as:

$$q(x_t|x_{t-1}, y) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1} + (1 - \sqrt{1-\beta_t})y, \beta_t I) \tag{18}$$

$$q(x_t|x_0, y) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0 + y, (1-\bar{\alpha}_t)I) \quad \text{where} \quad t > 1 \tag{19}$$

The posterior distribution $q(x_{t-1}|x_t, x_0)$ will also be a Gaussian distribution with the following form:

$$q(x_{t-1}|x_t, x_0, y) = \mathcal{N}(\hat{\mu}_t(x_t, x_0, y), \hat{\beta}_t I) \tag{20}$$

$$\text{where} \quad \hat{\mu}_t(x_t, x_0, y) = \underbrace{\frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}x_0 + \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t}x_t}_{\tilde{\mu}_t(x_t, x_0)} + \underbrace{(1 - \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t})}_{\kappa} y \tag{21}$$

$$\hat{\beta}_t = \frac{(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\beta_t \quad \text{and} \quad x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - y - \epsilon\sqrt{1-\bar{\alpha}_t}) \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, I) \tag{22}$$

Hence the forward process posterior in the VAEDM formulation is a shifted version of the forward process posterior proposed in [12]. Since the VAE reconstruction y for an image x_0 is constant during DDPM training, we can parameterize the reverse process posterior as $\hat{\mu}_\theta(x_t, x_0, y, t) = \tilde{\mu}_\theta(x_t, x_0, t) + \kappa y$. Additionally, we keep the variance of the reverse process conditional fixed and equal to $\hat{\beta}_t$ as proposed in [12]. Since $L_{t-1} \propto \|\hat{\mu}_t(x_t, x_0, y) - \hat{\mu}_\theta(x_t, x_0, y, t)\|^2$, the DDPM training objective in our formulation remains unchanged from the simplified denoising score matching objective proposed in [12].

Choice of the decoder, L_0: One possible choice for the decoder is to set $p_\theta(x_0|x_1)$ to be a discrete independent decoder derived from the Gaussian $\mathcal{N}(\hat{\mu}_\theta(x_1, y, 1), \hat{\beta}_1 I)$ [12]. However, at $t = 1$, we have $\hat{\mu}_\theta(x_1, y, 1) = x_0(x_1, y, \epsilon_\theta) + y$. Therefore, to account for the VAE reconstruction bias in the final DDPM output, we set our decoder $p_\theta(x_0|x_1) = \mathcal{N}(\hat{\mu}_\theta(x_1, y, 1) - y, \hat{\beta}_1 I)$. Without using this adjustment, we found the final DDPM samples to be a bit blurry in our initial experiments.The final training and inference algorithms are summarized in Algorithms 1 and 2 respectively. In Algorithm 2, the coefficients γ_0, γ_1 and γ_2 denote the coefficients of the forward process posterior in Eqn. 21.

B Appendix: Detailed Proofs

B.1 Derivation of VAEDM ELBO objective

As outlined in Section 3.1, the ELBO for the VAEDM log-likelihood can be specified as follows:

$$\log p(x_0) \geq \mathbb{E}_{q(x_{1:T},y,z|x_0)}[\log \frac{p(x_{0:T},y,z)}{q(x_{1:T},y,z|x_0)}] \tag{23}$$

where $p(x_{0:T},y,z)$ is the generative process and $q(x_{1:T},y,z|x_0)$ represents the joint posterior over the unknowns. Moreover the generative process and the joint posterior can be factorized as follows:

$$p(x_{0:T},y,z) = p(z)p_\theta(y|z)p_\phi(x_{0:T}|y,z) \tag{24}$$

$$q(x_{1:T},y,z|x_0) = q(y|x_0)q_\psi(z|y,x_0)q(x_{1:T}|y,z,x_0) \tag{25}$$

where θ, ϕ and ψ are the parameters of the VAE decoder, DDPM reverse process decoder and VAE recognition network, respectively. Plugging the factorial forms of the VAEDM generative process and the joint posterior defined above in eqn. (23), we can simplify the ELBO as follows:

$$\log p(x_0) \geq \mathbb{E}_{q(x_{1:T},y,z|x_0)}[\log \frac{p(x_{0:T},y,z)}{q(x_{1:T},y,z|x_0)}] \tag{26}$$

$$\geq \mathbb{E}_{q(x_{1:T},y,z|x_0)}[\log \frac{p(z)p_\theta(y|z)p_\phi(x_{0:T}|y,z)}{q(y|x_0)q_\psi(z|y,x_0)q(x_{1:T}|y,z,x_0)}] \tag{27}$$

$$\geq \mathbb{E}_{q(x_{1:T},y,z|x_0)}[\log \frac{p(z)}{q_\psi(z|y,x_0)} + \log p_\theta(y|z) + \log \frac{p_\phi(x_{0:T}|y,z)}{q(x_{1:T}|y,z,x_0)} - \log q(y|x_0)] \tag{28}$$

$$\geq \mathbb{E}_{q(y,z|x_0)}[\log \frac{p(z)}{q_\psi(z|y,x_0)} + \log p_\theta(y|z)] + \mathbb{E}_{q(x_{1:T},y,z|x_0)}[\log \frac{p_\phi(x_{0:T}|y,z)}{q(x_{1:T}|y,z,x_0)}] - \mathbb{E}_{q(y|x_0)}[\log q(y|x_0)] \tag{29}$$

$$\begin{aligned} \geq\ & \mathbb{E}_{q(y|x_0)}[\mathbb{E}_{q_\psi(z|y,x_0)} \log p_\theta(y|z) - \mathcal{D}_{KL}(q_\psi(z|y,x_0)\|p(z))] + \\ & \mathbb{E}_{q(y,z|x_0)}[\mathbb{E}_{q(x_{1:T}|y,z,x_0)}[\log \frac{p_\phi(x_{0:T}|y,z)}{q(x_{1:T}|y,z,x_0)}]] + \mathbb{H}[q(y|x_0)] \end{aligned} \tag{30}$$

B.2 Derivation of the forward conditional marginal in Formulation 2

Given:

$$q(x_1|x_0,y) = \mathcal{N}(\sqrt{1-\beta_1}x_0 + y, \beta_1 I) \tag{31}$$

$$q(x_t|x_{t-1},y) = \mathcal{N}(\sqrt{1-\beta_t}x_{t-1} + (1-\sqrt{1-\beta_t})y, \beta_t I) \tag{32}$$

From Eqn.(32), we can write,

$$x_t = \sqrt{1-\beta_t}x_{t-1} + (1-\sqrt{1-\beta_t})y + \epsilon\sqrt{\beta_t}, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0,I) \tag{33}$$

Taking expectations both sides,

$$\mathbb{E}(x_t) = \sqrt{1-\beta_t}\mathbb{E}(x_{t-1}) + (1-\sqrt{1-\beta_t})y \tag{34}$$

$$\mathbb{E}(x_t) = \sqrt{1-\beta_t}[\sqrt{1-\beta_{t-1}}\mathbb{E}(x_{t-2}) + (1-\sqrt{1-\beta_{t-1}})y] + (1-\sqrt{1-\beta_t})y \tag{35}$$

$$\mathbb{E}(x_t) = \sqrt{(1-\beta_t)(1-\beta_{t-1})}\mathbb{E}(x_{t-2}) + (1-\sqrt{(1-\beta_t)(1-\beta_{t-1})})y \tag{36}$$

$$\vdots \tag{37}$$

$$\mathbb{E}(x_t) = \sqrt{\prod_{t=2}^{t}(1-\beta_t)}\mathbb{E}(x_1) + y(1-\sqrt{\prod_{t=2}^{t}(1-\beta_t)}) \tag{38}$$

Substituting $\mathbb{E}(x_1) = \sqrt{1-\beta_1}x_0 + y$ from Eqn.(31) into the above formulation we get,

$$\mathbb{E}(x_t) = \sqrt{\prod_{t=1}^{t}(1-\beta_t)}x_0 + y = \sqrt{\bar{\alpha}_t}x_0 + y \tag{39}$$

Similarly it can be shown that $Var(x_t) = (1-\bar{\alpha}_t)I$. Therefore,

$$q(x_t|x_0, y) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0 + y, (1-\bar{\alpha}_t)I) \tag{40}$$

C Model Architecture and training

Unless otherwise mentioned, we re-used the model architectures and training hyperparameters proposed in [12]. For all experiments, we used the EMA approach from [10] for updating the target model parameters with an initial decay rate of 0.9999. However, we did not experiment with the choice of the EMA scheme.

Data preprocessing: For experiments with the CelebAMask-HQ dataset [22], we downsampled the training data to 128 x 128 resolution since we believe this setting provides the correct balance between visual details and training efficiency. We did not use any form of data augmentation during training for both CelebAMask-HQ and CIFAR-10 datasets. For Stage-1 VAE training, the training data was scaled between [0.0, 1.0]. For Stage-2 DDPM training, the images were scaled between [-1.0, 1.0] for training both the unconditional DDPM and the VAEDM (formulation-1) and between [0, 1.0] for the VAEDM (formualation-2) model.

Model architecture: The VAE architecture used for Stage-1 training on the CelebAMask-HQ dataset consists of around 21M parameters with residual block architectures inspired from [4]. The VAE latent code size was set to 1024 for CelebAMask-HQ and 512 for the CIFAR-10 dataset. The U-Net [31] decoder used in Stage-2 DDPM training is around 113M parameters. For CIFAR-10, the VAE model consists of around 9M parameters whereas the size of the U-Net decoder used is 34.8M parameters.

Training: The VAE model was trained for around 180 epochs and 500 epochs for the CelebAMask-HQ and the CIFAR-10 datasets with batch sizes of 40 and 128 respectively. All DDPM models were trained for around 800 epochs with a batch size of 64 for the CelebAMask-HQ dataset and for around 1000 epochs with a batch size of 128 for the CIFAR-10 dataset. We used a mix of 4 Nvidia 1080Ti GPUs (44GB memory) and a cloud TPUv2-8 (64GB memory) provided by Colab Pro for training the models. For DDPM training, it took around 0.74s and 0.2s per optimization step for the CelebAHQ and CIFAR-10 datasets respectively.

D Unconditional samples generated using VAEDM

Figure 7: Selected unconditional samples (128 x 128) generated from VAEDM using formulation 2. (t=500)

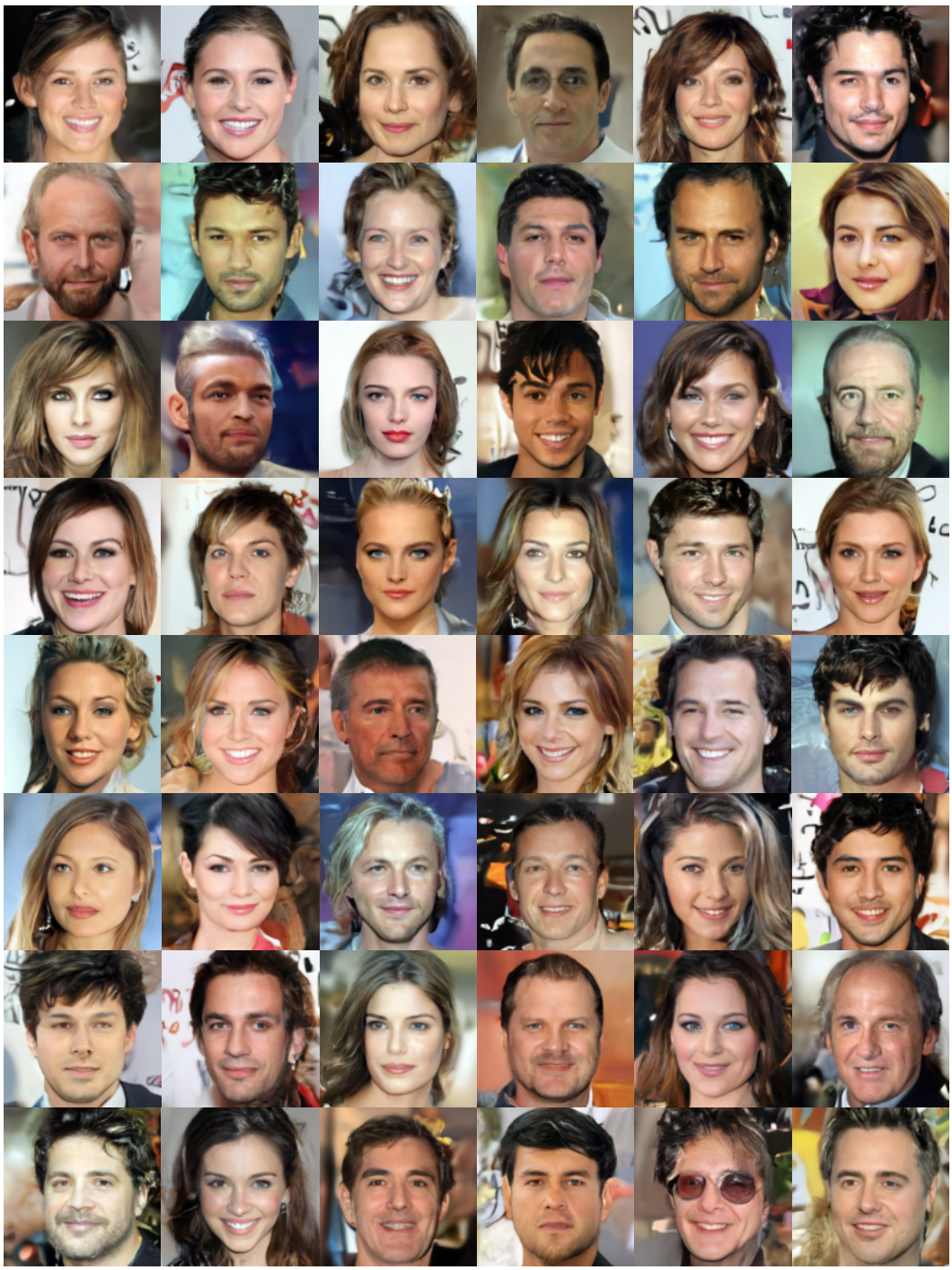

Figure 8: Selected unconditional samples (128 x 128) generated from VAEDM using formulation 1. (t=500)