Abstract

In this paper, we show that a binary latent space can be explored for compact yet expressive image representations. We model the bi-directional mappings between an image and the corresponding latent binary representation by training an auto-encoder with a multivariate Bernoulli encoding distribution. On the one hand, the binary latent space provides a compact discrete image representation of which the distribution can be modeled more efficiently than pixels or continuous latent representations. On the other hand, we now represent each image patch as a binary vector instead of an index of a learned cookbook as in discrete image representations with vector quantization. In this way, we obtain binary latent representations that allow for better image quality and high-resolution image representations without any multi-stage hierarchy in the latent space. In this binary latent space, images can now be generated effectively using a binary latent diffusion model tailored specifically for modeling the prior over the binary image representations. We present both conditional and unconditional image generation experiments with multiple datasets, and show that the proposed method performs comparably to state-of-the-art methods while dramatically improving the sampling efficiency to as few as 16 steps without using any test-time acceleration. The proposed framework can also be seamlessly scaled to 1024 × 1024 high-resolution image generation without resorting to latent hierarchy or multi-stage refinements.

1. Introduction

The goal of modeling the image distribution that allows efficient generation of high-quality novel samples drives the research of representation learning and generative models. Directly representing and generating images in the pixel space stimulates various research such as generative adversarial networks [2, 7, 18, 28], flow models [11, 33, 41, 45], energy-based models [12, 13, 64, 69], and diffusion models [24, 39, 52, 53]. As the resolution grows, it becomes increasingly difficult to accurately regress the pixel values. And this challenge usually has to be addressed through hierarchical model architectures [28, 70] or at a notably high cost [24]. Moreover, while demonstrating outstanding generated image quality, GAN models suffer from issues including insufficient mode coverage [36] and training instability [21].

Representing and generating images in a learned latent space [32, 40, 47] provides a promising alternative. Latent diffusion [47] performs denoising in a latent feature space with a lower dimension than the pixel space, therefore reduces the cost of each denoising step. However, regressing the real-value latent representations remains complex and demands hundreds of diffusion steps. Variational auto-encoders (VAEs) [23, 32, 46] generate images without any iterative steps. However, the static prior of the latent space re-
stricts the expressiveness, and can lead to posterior collapse. To achieve higher flexibility of the latent distribution without significantly increasing the modeling complexity, VQ-VAE [59] introduces a vector-quantized latent space, where each image is represented as a sequence of indexes, each of which points to a vector in a learned codebook. The prior over the vector-quantized representations is then modeled by a trained sampler, which is usually parametrized as an autoregressive model. The success of VQ-VAE stimulates a series of works that model the discrete latent space of code-book indexes with different models such as accelerated parallel autoregressive models [8] and multinomial diffusion models [6, 20]. VQ-based generative models demonstrate surprising image synthesis performance and model coverage that is better than the more sophisticated methods like GANs without suffering from issues like training instability. However, the hard restriction of using one codebook index to represent each image patch introduces a trade-off on the codebook size, as a large enough codebook to cover more image patterns will introduce an over-complex multivariate multinomial latent distribution for the sampler to model.

In this research, we explore a compact yet expressive representation of images in a binary latent space, where each image patch is now represented as a binary vector, and the prior over the discrete binary latent codes is effectively modeled by our improved binary diffusion model tailored for multivariate Bernoulli distribution. Specifically, the bi-directional mappings between images and the binary representations are modeled by a feed-forward autoencoder with a binary latent space. Given an image, the encoder now outputs the normalized parameters of a multivariate Bernoulli distribution, from which a binary representation is constructed in our method in Figure 1. We support our findings with both conditional and unconditional image generation experiments on multiple datasets. We show that our method can deliver remarkable image generation quality and diversity with more compact latent codes, larger image-to-latent resolution ratios, as well as fewer sampling steps and faster sampling speed. We present some examples with different resolutions generated by the proposed method in Figure 1.

We organize this paper as follows: Related works are discussed in Section 2. In Section 3, we introduce binary image representations by training an auto-encoder with a binary latent space. We then introduce in Section 4 binary latent diffusion, a diffusion model for multivariate Bernoulli distribution, and techniques tailored specifically for improving the training and sampling of binary latent diffusion. We present both quantitative and qualitative experimental results in Section 5 and conclude the paper in Section 6.

2. Related Work

Diffusion models and image generation. Diffusion models [52] are proposed as a flexible way of modeling complex data distribution using tractable families of probability distributions. Denoising diffusion probabilistic models (DDPM) [24] build a reparameterization of the learning objective that connects diffusion probabilistic models and denoising score matching [54]. The resulting diffusion models based on multivariate Gaussian distributions scale up the generation resolution and achieve high-quality image generation with quality comparable with strong generative families such as GANs [7, 30, 37], score-based models [54–56], and energy-based models [14, 19], without suffering from issues like mode collapse and training instability. Further improvements have been proposed to encourage higher log-likelihoods [39], reduce computational demands [47], and extend diffusion models to broader applications including text-to-image generation [42, 48], planning [27], super-resolution [49], temporal data modeling [1, 34, 58], and adversarial robustness [5, 61]. Particularly, diffusion models with discrete states are advocated in [3] for text generation. A series of structured transition matrices are introduced in [3], among which diffusion with an absorbing state is further extended to image generation with VQ codes in [6].

Discrete representations. Modern deep neural networks trained with back-propagation and gradient descent prevalently advocate continuous features across all layers. However, discrete representations still demonstrate their unique advantages and applications. [4] discusses several approaches to estimating the gradient through stochastic neu-
rons. Among the four approaches introduced in [4], the practically simplest one with heuristic gradient copies is further adopted in learning fully discrete vector-quantized auto-encoders [59], which builds the foundations for many follow-up methods [6, 8, 16, 44]. [9] proposes rehearsing training samples for continual learning in a binary latent space. [17] introduces a binary latent space by a hard threshold instead of sampling in our methods. And the sampling with random hyperplane rounding in [47] can hardly be scaled to a higher resolution.

3. Binary Image Representations

Given an image dataset, we begin with learning the bi-directional mappings between images and their binary representations. This is achieved by training an auto-encoder with a binary latent space, where the binary code of each image is obtained as a sample of a multivariate Bernoulli distribution inferred from the image. Specifically, denoting an image as \( x \in \mathbb{R}^{h \times w \times 3} \), we train an image encoder \( \Psi \) that outputs the unnormalized parameters for the corresponding multivariate Bernoulli distribution \( \Psi(x) \). A Sigmoid non-linearity \( \sigma \) is then adopted to normalize the parameters \( y \in \mathbb{R}^{h \times w \times c} = \sigma(\Psi(x)) \), where \( h \) and \( w \) denote the spatial resolution of the image, \( k \) is the downsampling factor of the encoder \( \Psi \), and \( c \) is the number of encoded feature channels.

To obtain the binary representations of images, we perform Bernoulli sampling given the normalized parameters \( z = \text{Bernoulli}(y) \). Note that the Bernoulli sampling operation here does not naturally permit gradient propagation through it, and prevents the end-to-end training of the encoder-decoder architecture. In practice, we consistently observe that a straight-through gradient estimation by direct copying the gradients and skipping the non-differentiable sampling in backpropagation can maintain both stable training and superior performance. The straight-through gradient estimation can be easily implemented by a surrogate function as:

\[
\tilde{z} = \odot(z) + y - \odot(y),
\]

where \( \odot(\cdot) \) denotes the stop gradient operation. \( \tilde{z} \), which is binary and identical to \( z \), will be sent to the following decoder network \( \Phi \) for image reconstruction. The gradient propagated back from the decoder \( \Phi \) to \( \tilde{z} \) is directly sent to \( y \), which is differentiable w.r.t. the encoder \( \Psi \) and permits the end-to-end training of the entire auto-encoder with a discrete binary latent space.

The reconstruction of the image is obtained using a decoder network \( \Phi \) as \( \hat{x} = \Phi(\tilde{z}) \). The overall framework of the binary autoencoder is visualized in Figure 2. With the gradient surrogate function ensuring the end-to-end gradient propagation, the network is trained by minimizing the final objective

\[
\mathcal{L} = \sum_{i=1}^{||c||} \omega_i C[i](\hat{x}, x),
\]

where \( C \) denotes a collection of loss functions such as mean squared error, perception loss, and adversarial loss. And \( \omega_i \) denotes the weights that balance each loss term.

4. Multivariate Bernoulli Diffusion

Given an image dataset and the corresponding binary latent representations of each image, we then discuss how to effectively model the prior over the binary latent codes with a parametrized model \( p_\theta(z) \), from which novel samples of the binary latent codes can be efficiently sampled. To do so, we introduce binary latent diffusion, a diffusion model tailored specifically for multivariate Bernoulli distribution accompanied with improvement techniques that promote stable and effective training and sampling.

A diffusion model is usually established by first defining a \( T \)-step diffusion process consisting of a sequence of variation distributions \( q(z|z^{-1}) \), with \( t \in \{1, \ldots, T\} \). Each variation distribution in \( q(z|z^{-1}) \) is defined to progressively add noise to \( z^{-1} \), so that with sufficient steps \( T \) and a valid noise scheduler defining \( q(z|z^{-1}) \), the final state \( q(z^T|z^0) \) converges to a known random distribution that is easy to evaluate and sample from, and conveys almost no valid information of \( z^0 \). Specifically, in our case, we are interested in modeling the prior distribution of the binary latent code of images. Therefore, we define the starting point of the diffusion process with our latent code distribution \( z \sim q(z^0) \), where \( q(z^0) \) is a multivariate Bernoulli distribution as the prior of latent codes. And the full diffusion process \( q \) can be defined as

\[
q(z^{1:T}) := \prod_{t=1}^{T} q(z_t|z^{t-1}), \quad \text{with} \quad q(z_t|z^{t-1}) = B(z^{t-1}, 1 - \beta^t + 0.5\beta^t),
\]

where \( B \) denotes a multivariate Bernoulli distribution and \( \beta^t \) defines the noise scale at each step \( t \). With sufficient steps \( T \), and \( \beta^t \) at adequate scales, the forward diffusion process will converge to \( B(z^T, 0.5) \), which is a random Bernoulli distribution that is easy to sample from. Figure 3 illustrates an example of the binary latent diffusion process.

Given an arbitrary time step \( t \) and a sample \( z^0 \), the posterior distribution can be easily obtained as:

\[
q(z^{t}, z^{t-1}; z^0) = B(z^{t-1}, 1 - \beta^t + 0.5\beta^t), \quad \text{with} \quad k^t = \prod_{i=1}^{t} (1 - \beta^i), \quad b^t = (1 - \beta^t)b^{t-1} + 0.5\beta^t, \quad \text{and} \quad b^1 = 0.5\beta^1,
\]
where $k^t$ and $b^t$ here jointly define the accumulated noise scale till time step $t$. The Markov chain and the analytic form of the posterior in (5) permit stochastic sampling in each training batch.

The noise schedulers of the proposed binary latent diffusion process can be constructed by either simply defining the noise scale $\beta^t$ at each step, or directly defining the accumulated noise factors $k^t$ and $b^t$. Note that even with $k^t$ and $b^t$ directly defined, the corresponding $\beta^t$ can still be obtained as $\beta^t = 1 - \frac{k^t}{k^T}$. We present in Appendix Section A discussions on different choices of noise schedulers.

With the forward diffusion process properly defined, the goal now is to train a function $f_\theta$ with $z^{T-1} = f_\theta(z^t, t)$ to model the reverse diffusion (denoising) process

$$p_\theta(z^{T-1}|z^t) = q(z^{T-1}; f_\theta(z^t, t)), \quad (6)$$

which allows sampling to be performed by reversing a sample from $q(z^T)$ to a sample in $q_\theta(z^0)$. The diffusion process $q$ and the denoising process $p$ jointly define a variational auto-encoder [32], with the variational lower bound:

$$L_{vib} := \mathcal{L}_0 + \sum_{t=1}^{T-1} \mathcal{L}_t + \mathcal{L}_r$$

$$:= -\log p_\theta(z^0|z^t)$$

$$+ \sum_{t=1}^{T-1} KL(q(z^{T-1}|z^t, z^0)||p_\theta(z^{T-1}|z^t))$$

$$+ KL(q(z^T||p(z^T))). \quad (7)$$

Note that in this paper we focus on the diffusion process with fixed predefined noise scheduler $\beta^t$, therefore the third term on the RHS of (7) does not depend on $\theta$ and is always nearly 0. All the KL divergence and likelihood terms in (7) can be analytically calculated in a closed form as all distributions involved are multivariate Bernoulli.

### 4.1. Binary Latent Diffusion Reparameterization

To learn the reverse diffusion process, a straightforward way is to train a neural network $f_\theta$ using (7) to model $p_\theta(z^{T-1}|z^t)$. According to (5), each $z^t$ can be considered as a linear interpolation between $z^0$ and $z^T$, whose parameters can take arbitrary numbers between 0 and 1 depending on the noise scheduler. Therefore, directly training $f_\theta$ to model $p_\theta(z^{T-1}|z^t)$ can be challenging as the model needs to accurately regress such sophisticated interpolations.

**Predicting $z^0$.** Inspired by [20], we first introduce a reparameterization to the prediction target as $p_\theta(z^0|z^t)$ and train the model $f_\theta$ to directly predict $z^0$ as $z^0 = f_\theta(z^t, t)$ at each step. During sampling, at each step, with $p_\theta(z^0|z^t)$ predicted, we can recover the corresponding $p_\theta(z^{T-1}|z^t)$ by:

$$p_\theta(z^{T-1}|z^t) = q(z^{T-1}|z^t, z^0 = 0)p_\theta(z^0 = 0|z^t)$$

$$+ q(z^{T-1}|z^t, z^0 = 1)p_\theta(z^0 = 1|z^t), \quad (8)$$

with

$$q(z^{T-1}|z^t, z^0) = \frac{q(z^t|z^{T-1}, z^0)q(z^{T-1}|z^0)}{q(z^t|z^0)}. \quad (9)$$

Specifically, with a binary code $z^t$ and the noise scheduler
defined in (5), we have
\[
p_\theta(z_t^{-1}|z_t) = B(z_{t-1}^-|[(1 - \beta_t^t)z_t^t + 0.5\beta_t^t] \odot [k^t f_\theta(z_t^t, t) + 0.5b_t^t]),
\]
where
\[
Z = [(1 - \beta_t^t)z_t^t + 0.5\beta_t^t] \odot [k^t f_\theta(z_t^t, t) + 0.5b_t^t]
\]
+ [(1 - \beta_t^t)(1 - z_t^t) + 0.5\beta_t^t] \odot [k^t(1 - f_\theta(z_t^t, t)) + 0.5b_t^t]
\] 
(11)
is a normalization term that guarantees valid probabilities in (10), and \(\odot\) denotes element-wise product. With the introduced reparameterization, the prediction targets at all time steps become \(z^0\), which is strictly binary, and eases the training according to our observations.

**Predicting the residual.** A reparameterization to the prediction target as the residual is introduced in [24], which bridges the connections between diffusion and denoising score matching [54] and improves the sampling results. In the proposed binary latent diffusion, the decreasing noise scales result in \(z^t\) getting closer to \(z^0\) as \(t\) decreases. Thus, the sampling in the binary latent space naturally favors fewer flippings to the binary codes as \(t\) gets closer to \(t = 0\). To better capture this sparsity in predictions and stabilize the sampling to prevent divergence, we show that, while predicting the residual is considered non-trivial for discrete-state diffusion models [26], the prediction targets can be further reparameterized as the residual between \(z^0\) and \(z^t\) in our binary latent diffusion. Specifically, we train \(f_\theta(z_t^t, t)\) to fit \(z_t^t \oplus z^0\), where \(\oplus\) denotes the element-wise logic XOR operation. The model \(f_\theta\) is now trained to predict the ‘flipping probability’ of the binary code. And the prediction targets remain strictly binary.

**Final training objective.** The final simplified training objective can be formulated as:
\[
\mathcal{L}_{\text{residual}} = \mathbb{E}_{c, z^0} \text{BCE}(f_\theta(z^t, t), z^t \oplus z^0),
\]
(12)
where \(\text{BCE}(\cdot, \cdot)\) denotes the binary cross-entropy loss. In practice, following [39], we find it beneficial to set the final learning objective as a combination of \(\mathcal{L}_{\text{residual}}\) and \(\mathcal{L}_{\text{vlb}}\) as
\[
\mathcal{L} = \mathcal{L}_{\text{residual}} + \lambda \mathcal{L}_{\text{vlb}},
\]
(13)
with \(\lambda\) is a small number.

**Sampling temperatures.** In practice, the prediction of the residual \(f_\theta(z_t^t, t)\) is implemented as \(f_\theta(z_t^t, t) = \sigma(T_\theta(z_t^t, t))\), where \(T_\theta\) is a plain transformer that outputs the unnormalized flipping probability, which is then normalized by a Sigmoid function \(\sigma\). During sampling, we can manually adjust the sampling diversity by inserting a temperature \(\tau\) which turns the prediction to \(f_\theta(z_t^t, t) = \sigma(T_\theta(z_t^t, t)/\tau)\). Note that \(\tau\) is only used to adjust the diversity during post-training sampling, and is not included as a hyperparameter in training. Examples of how \(\tau\) affects the sample diversity are shown in Figure 4.

**Classifier-free guidance.** Classifier-free guidance [25] is introduced to improve the generation fidelity and promote high correspondence to the conditions for conditional diffusion models. While it is previously believed that classifier-free guidance can hardly be extended to diffusion models, we show that the reparameterization to the prediction target as the binary residuals allows classifier-free guidance to apply seamlessly to the propose binary latent diffusion for improved image fidelity in conditional image generation. In conditional image generation, we introduce an extra condition token \(c\) that carries the conditions such as class embedding in ImageNet class-conditional image generation or text embedding in the text-to-image generation.

The model performs class-conditional prediction as \(f_\theta(z_t^t, t, c) = \sigma(T_\theta(z_t^t, t, c))\). An unconditional prediction can be drawn by simply dropping the condition token \(c\) as \(f_\theta(z_t^t, t) = \sigma(T_\theta(z_t^t, t))\). The final prediction at each step with classifier-free guidance can then be implemented as
\[
f_\theta(z_t^t, t, c) = \sigma((1 + \omega)T_\theta(z_t^t, t, c) - \omega T_\theta(z_t^t, t)),
\]
where \(\omega\) is a non-negative scalar controlling the strength of the guidance. Note that same the temperature scale \(\tau\), the guidance strength \(\omega\) is only effective in the sampling stage and is not involved in training, which allows as to arbitrarily adjust the sampling quality without retraining the models. While it is inevitable that the classifier-free guidance doubles that computation cost at each sampling step due to the extra unconditional predictions, we consistently observe performing classifier-free guidance allows us to skip sampling steps. For example, performing only a quarter of the sampling steps enhanced with classifier-free guidance reduce the overall computation cost by half, and does not noticeably reduce the sample quality. We present in Appendix Figure M results generated with different scales of classifier-free guidance.
4.2. Comparisons with Existing Methods

In this section, we briefly discuss the advantages of adopting the proposed binary representations over other alternatives of latent space image representations. As visualized in Figure 5, vector-quantized latent space [16, 59] represents each image patch as a discrete index, or equivalently, a one-hot vector. The one-hot vector then multiplies with the learned codebook to obtain the feature representation of an image patch. Image representations in continuous latent space [47] can be interpreted in a similar way by simply treating the first weight matrix in the decoder network as the learned codebook. The real-value latent code of an image patch performs arbitrary linear combinations of vectors in the codebook and is able to cover diverse feature space even with a low-dimensional code for higher efficiency. Our method with binary representation strives for a balance between these two methods by restricting the vectors composing the codebook to be binary. On the one hand, the binary composition of a codebook offers a much more diverse and flexible composition of features compared to the vector-quantized representations. For example, a compact 32-bit binary vector can represent over 4,000 million patterns, which is much larger than the 1,024 patterns commonly used in vector-quantized representations [6, 8, 32]. As we will further show empirically in Section 5, this improved coverage of patterns allows for higher expressiveness and enables high-resolution image generation that can hardly be accomplished by VQ representations at high quality. On the other hand, the binary restriction guarantees that the representations remain compact. As we will show in Section 5.3, our 8k-bit binary representations perform comparably with a latent diffusion model [47] with 131k-bit representations.

5. Experiments

In this section, we present both unconditional and conditional image generation experiments with multiple datasets. Following common practice [6, 8], we train a plain transformer network [60] to parametrize the sampler $T_\theta$. We use a temperature of $\tau = 0.9$ as the default setting for sampling, and $\lambda = 0.1$ in all the experiments, if not otherwise specified. For the number of channels in the binary latent space, we use $c = 32$ for $256 \times 256$ unconditional image generation, $c = 64$ for class-conditional image generation, and $1024 \times 1024$ high-resolution unconditional image generation. Additional details regarding the implementation and evaluation metrics can be found in Appendix Section B.

5.1. Unconditional Image Generation

In this section, we present results and comparisons on unconditional image generation experiments with multiple datasets including LSUN Bedrooms and Churches [66], FFHQ [29], and CelebA-HQ [28]. To allow for fair comparisons, we first present $256 \times 256$ image generation. We apply our method on a $16 \times 16 \times 32$ latent space, which corresponds to the same $16 \times$ downsampling of the spatial resolution as in methods like [6, 8]. Note that as we further show in Section 5.3, the improved expressiveness of our binary representation compared to the vector-quantized representation allows our method to perform well with a larger downsampling ratio such as $32 \times$. Each image patch is now represented with a 32-bit binary vector, and an image is represented as an 8k-bit tensor, which is more compact compared to the 131k-bit representations in Latent Diffusion models [47].

For comprehensive evaluations and comparisons, we adopt Fréchet Inception Distance (FID) and precision-recall [35] as the evaluation metrics. We present comparisons against various state-of-the-art methods in Table 1. We conduct comparisons by generating 50,000 samples and comparing them with the corresponding training dataset in every experiment. Quantitative comparisons against state-of-the-art methods are presented in Table 1. Despite using much fewer denoising steps compared to latent diffusion models (LDM) [47] and absorbing diffusion models [6], our binary latent diffusion models achieve comparable image generation quality with desirable diversity. Note that our results are obtained without any test-time accelerations or skipping denoising steps. While LDM [47] results are reported with
Table 1. Quantitative comparisons on unconditional image generation with LSUN Bedrooms, LSUN Churches, FFHQ, and CelebA-HQ. We also provide comparisons on the number of denoising steps in training. All results are obtained with a resolution of $256 \times 256$.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Steps</th>
<th>Bedrooms</th>
<th>Churches</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>FID ↓ P ↑ R ↑</td>
<td>FID ↓ P ↑ R ↑</td>
</tr>
<tr>
<td>DCT [38]</td>
<td>-</td>
<td>6.40 0.44 0.55</td>
<td>7.56 0.60 0.48</td>
</tr>
<tr>
<td>ImageBART [15]</td>
<td>5.51</td>
<td>-        -</td>
<td></td>
</tr>
<tr>
<td>VQGAN [16]</td>
<td>256</td>
<td>6.35 0.61 0.33</td>
<td>7.81 0.67 0.29</td>
</tr>
<tr>
<td>DDPM [24]</td>
<td>1,000</td>
<td>6.36 - -</td>
<td>7.89 - -</td>
</tr>
<tr>
<td>PGGAN [28]</td>
<td>-</td>
<td>8.34 0.43 0.40</td>
<td>6.42 0.61 0.38</td>
</tr>
<tr>
<td>StyleGAN [29, 30]</td>
<td>-</td>
<td>2.35 0.55 0.48</td>
<td>3.86 0.60 0.43</td>
</tr>
<tr>
<td>Absorbing [6]</td>
<td>256</td>
<td>3.64 0.67 0.38</td>
<td>4.07 0.71 0.45</td>
</tr>
<tr>
<td>LDM [47]</td>
<td>1,000</td>
<td>2.95 0.66 0.48</td>
<td>4.02 0.64 0.52</td>
</tr>
<tr>
<td>Ours</td>
<td>16</td>
<td>4.11 0.62 0.44</td>
<td>4.66 0.66 0.48</td>
</tr>
<tr>
<td>Ours</td>
<td>64</td>
<td>3.85 0.65 0.44</td>
<td>4.36 0.68 0.50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VQGAN</td>
<td>400</td>
<td></td>
<td>10.2 - -</td>
</tr>
<tr>
<td>StyleGAN [29]</td>
<td>1</td>
<td>4.16 0.71 0.46</td>
<td></td>
</tr>
<tr>
<td>LDM [31]</td>
<td>1,000</td>
<td>5.52 - -</td>
<td>7.16 - -</td>
</tr>
<tr>
<td>Absorbing [6]</td>
<td>256</td>
<td>6.11 0.73 0.48</td>
<td>- - -</td>
</tr>
<tr>
<td>LDM [47]</td>
<td>1,000</td>
<td>4.98 0.73 0.50</td>
<td>5.11 0.72 0.49</td>
</tr>
<tr>
<td>Ours</td>
<td>16</td>
<td>6.48 0.71 0.45</td>
<td>7.80 0.67 0.44</td>
</tr>
<tr>
<td>Ours</td>
<td>64</td>
<td>5.85 0.73 0.50</td>
<td>6.24 0.71 0.48</td>
</tr>
</tbody>
</table>

5.2. Conditional Image Generation

We adopt the ImageNet-IK dataset for image generation with class labels as conditions. In addition to FID and precision-recall, we follow the common practice and include the inception score (IS) [50] and the classification accuracy score (CAS) [43] with ResNet-50 [22] as the evaluation metrics. Following the same settings as in the unconditional image generation experiments, we generate images at a resolution of $256 \times 256$, with a latent space of $16 \times 16 \times 64$. We present quantitative comparisons against state-of-the-art methods in Table 2. Note that the results of our method are achieved without any auxiliary sampling strategies or accelerations such as classifier [10] or classifier-free guidance [25]. Our results demonstrate comparable image quality to state-of-the-art image generation methods and appealing sample diversity. The effectiveness is validated by the improved CAS scores. We present qualitative results in Figure 6 and Appendix Section C.3.

5.3. Discussions

**Image Reconstruction.** In this section, we present comparisons of the image reconstruction quality with different ways of formulating the latent space. To assess the reconstruction quality, we use the peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) [62] as the metrics to measure the reconstruction quality. We measure the performance on the validation set of the LSUN bedrooms dataset at a resolution of $256 \times 256$, and present the results in Table 4. Compared to VQ-based representation, our method permits better image reconstruction quality with a compact 32-bit binary representation for each image patch. The advantage is further amplified when a more compact spatial size ($8 \times 8$), which corresponds to a $32 \times$ downsampling ratio is adopted. The higher downsampling ratio allows our method to generate high-resolution images in a one-shot fashion as reported in Section 5.1.

**Alternative samplers.** To show the advantages of using the binary latent diffusion for the modeling of the prior over the multivariate Bernoulli latent distribution, we present in Table 5 quantitative results with the samplers parametrized by alternative architectures including autoregressive models as in [59] and absorbing diffusion [5]. We report FID results on $256 \times 256$ unconditional image generation with
LSUN Churches and FFHQ. The proposed binary latent diffusion model tailored for the multivariate Bernoulli distribution achieves the best results on modeling the prior over the binary latent image representations. Compared to autoregressive models and absorbing diffusion models which perform greedy sampling, our method allows the prediction at previous denoising steps to be modified and improved in later steps, and prevents error accumulation across steps.

**Efficiency.** One of the major advantages of the proposed efficient image generation with binary latent diffusion is that good results can be achieved with fewer steps of denoising, which results in a faster sampling speed in practice.

Table 2. Quantitative comparisons on $256 \times 256$ class-conditional image generation. We report both the top-1 and top-5 accuracy of CAS. Using the official training set obtains Top-1 and Top-5 accuracy of 76.6% and 93.1%, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Params</th>
<th>Step</th>
<th>FID ↓</th>
<th>IS ↑</th>
<th>P ↑</th>
<th>R ↑</th>
<th>Top-1 ↑</th>
<th>Top-5 ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCT [38]</td>
<td>738M</td>
<td>1024</td>
<td>36.51</td>
<td>-</td>
<td>0.36</td>
<td>0.67</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BigGAN-deep [7]</td>
<td>160M</td>
<td>1</td>
<td>6.95</td>
<td>198.2</td>
<td>0.87</td>
<td>0.28</td>
<td>43.99</td>
<td>67.89</td>
</tr>
<tr>
<td>Improved DDPM</td>
<td>280M</td>
<td>250</td>
<td>12.26</td>
<td>-</td>
<td>0.70</td>
<td>0.62</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ADM [10]</td>
<td>554M</td>
<td>250</td>
<td>10.94</td>
<td>101.0</td>
<td>0.69</td>
<td>0.63</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VQVAE2 [44]</td>
<td>13.5B</td>
<td>5120</td>
<td>31.11</td>
<td>~45</td>
<td>0.36</td>
<td>0.57</td>
<td>54.83</td>
<td>77.59</td>
</tr>
<tr>
<td>VQGAN [16]</td>
<td>1.4B</td>
<td>256</td>
<td>15.78</td>
<td>78.73</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MaskGIT [8]</td>
<td>227M</td>
<td>8</td>
<td>6.18</td>
<td>182.1</td>
<td>0.80</td>
<td>0.51</td>
<td>63.14</td>
<td>84.45</td>
</tr>
<tr>
<td>LDM [47]</td>
<td>400M</td>
<td>250</td>
<td>10.56</td>
<td>103.49</td>
<td>0.71</td>
<td>0.62</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ours</td>
<td>172M</td>
<td>64</td>
<td>8.52</td>
<td>158.1</td>
<td>0.72</td>
<td>0.64</td>
<td>65.07</td>
<td>85.60</td>
</tr>
</tbody>
</table>

Figure 6. Class-conditional image generation with classes: 218, 937, 973, 980.

Table 3. FID comparisons on $1024 \times 1024$ high-resolution image generation with FFHQ and CelebA-HQ. † indicates results we obtain based on the official implementations.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Steps</th>
<th>FID ↓</th>
<th>FFHQ ↓</th>
<th>CelebA-HQ ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>StyleSwin [68]</td>
<td>1</td>
<td>5.07</td>
<td>4.43</td>
<td></td>
</tr>
<tr>
<td>StyleGAN-XL [51]</td>
<td>1</td>
<td>2.02</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Diffusion StyleGAN2 [63]</td>
<td>≤1,000</td>
<td>2.83</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>LDM [47]</td>
<td>1,000</td>
<td>10.09</td>
<td>8.68</td>
<td></td>
</tr>
<tr>
<td>Absorbing † [6]</td>
<td>1,024</td>
<td>14.12</td>
<td>13.82</td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>64</td>
<td>6.53</td>
<td>5.73</td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>256</td>
<td>6.07</td>
<td>5.22</td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Image reconstruction quality with different formats of latent code. The sizes are formulated as height $\times$ width $\times$ code (or codebook) size.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Size</th>
<th>PSNR ↑</th>
<th>SSIM ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>VQ [59]</td>
<td>16x16x1024</td>
<td>20.08 ±1.8 ±0.62 ±0.09</td>
<td></td>
</tr>
<tr>
<td>Real value [47]</td>
<td>16x16x16</td>
<td>24.08 ±4.22 ±0.70 ±0.12</td>
<td></td>
</tr>
<tr>
<td>Binary (Ours)</td>
<td>16x16x32</td>
<td>24.02 ±2.11 ±0.75 ±0.09</td>
<td></td>
</tr>
<tr>
<td>VQ [59]</td>
<td>8x8x2048</td>
<td>17.20 ±1.45 ±0.53 ±0.09</td>
<td></td>
</tr>
<tr>
<td>Real value [47]</td>
<td>8x8x64</td>
<td>22.74 ±1.87 ±0.62 ±0.15</td>
<td></td>
</tr>
<tr>
<td>Binary (Ours)</td>
<td>8x8x128</td>
<td>22.82 ±1.84 ±0.64 ±0.07</td>
<td></td>
</tr>
</tbody>
</table>

Table 5. FID results with alternative samplers for the modeling of the prior over the multivariate Bernoulli latent distribution.

<table>
<thead>
<tr>
<th>Samplers</th>
<th>Steps</th>
<th>Churches ↓</th>
<th>FFHQ ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoregressive</td>
<td>256</td>
<td>7.25</td>
<td>8.23</td>
</tr>
<tr>
<td>Absorbing diffusion</td>
<td>256</td>
<td>5.44</td>
<td>7.64</td>
</tr>
<tr>
<td>Binary Diffusion (Ours)</td>
<td>64</td>
<td>4.36</td>
<td>5.85</td>
</tr>
</tbody>
</table>

Table 6. Comparisons on image generation speed with seconds per sample (s/sample). All results are obtained by averaging 1,000 times of sampling with a batch size of 1. 64s and 16s denote 64 and 16 denoising steps, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>StyleGAN-2</th>
<th>Absorbing</th>
<th>LDM</th>
<th>DDPM</th>
<th>Ours 64s</th>
<th>Ours 16s</th>
</tr>
</thead>
<tbody>
<tr>
<td>s/sample</td>
<td>0.04</td>
<td>3.40</td>
<td>15.68</td>
<td>63.85</td>
<td>0.82</td>
<td>0.20</td>
</tr>
</tbody>
</table>

We compare sampling speed against StyleGAN-2 [30], absorbing diffusion [6], latent diffusion [47], and DDPM [24] in Table 6. Our method demonstrates clear advantages in image sampling speed compared to other diffusion-based methods, and further closes the gap between the speed of GANs and diffusion models. We further present comparisons on the generation results with different denoising steps against absorbing diffusion models [6] in Figure 7. Our method demonstrates noticeably higher robustness to the number of steps, and fair results can be obtained with as few as 8 denoising steps. Meanwhile, we consistently observe that it is extremely hard to train Gaussian-based diffusion models such as latent diffusion models [47] and DDPM [24] to perform reasonably with fewer than 100 denoising steps in training.

**Ablation studies.** We present in Table 7 quantitative results that validate the values of $\lambda$ in (13) and the prediction targets of our binary latent diffusion. We consistently observe that the value of $\lambda$ correlates noticeably with the sample diversity, while a very large value of $\lambda$, such as $\lambda = 1.0$, fre-
Table 7. Performance (FID / Recall) with different values of $\lambda$ and prediction targets. All results are obtained on the LSUN Churches experiments.

<table>
<thead>
<tr>
<th>$\lambda$ values</th>
<th>$\lambda = 0$</th>
<th>$\lambda = 0.01$</th>
<th>$\lambda = 0.1$</th>
<th>$\lambda = 1.0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FID / R</td>
<td>5.07 / 0.42</td>
<td>4.68 / 0.46</td>
<td><strong>4.36 / 0.50</strong></td>
<td>5.12 / 0.48</td>
</tr>
<tr>
<td>Targets</td>
<td>$z^{l-1}$</td>
<td>$z^0$</td>
<td>$z^1 \oplus z^0$</td>
<td>-</td>
</tr>
<tr>
<td>FID / R</td>
<td>5.09 / 0.48</td>
<td>4.72 / 0.50</td>
<td><strong>4.36 / 0.50</strong></td>
<td>-</td>
</tr>
</tbody>
</table>

frequently causes sampling divergence, which results in samples that appear as random compositions of multiple images, and thus degrades the quality. Setting the prediction targets as $z^{l-1}$ corresponds to using only $L_{vlb}$ as the supervision. While different parameterizations to the prediction targets achieve similar sample diversity, the proposed repa-rameterization of the targets with the flipping probability $z^1 \oplus z^0$ achieves the best overall performance as it noticeably reduces the samples with strong artifacts caused by potentially diverged sampling according to our observations.

Image inpainting. Compared to the autoregressive-based samplers, one of the clear advantages of diffusion models is that the iterative sampling process in diffusion models does not assume any specific generation order of the spatial positions. This means that after training the generative models, we can generate images with partial observations as conditions. We present in Figure 8 examples of inpainting results generated by our models. Additional inpainting results can be found in Appendix Section C.4.

6. Conclusion

In this paper, we presented representing and generating images in a binary latent space. We learned binary image representations by training an auto-encoder with a multi-variate Bernoulli latent distribution and simple gradient copying to bypass the non-differential Bernoulli sampling operation. Compared to the real-value image representations in either the pixel or latent space, we showed that binary representation allows for a compact representation with the corresponding distribution that can be effectively modeled by a binary latent diffusion model. Compared to vector-quantized discrete representation, the binary representation in our work enables a higher expressiveness that permits high-resolution images to be generated without any multi-stage latent hierarchy. We examined our idea on multiple datasets with both conditional and unconditional image generation experiments. The comparable results to the recent state-of-the-art methods validated the effectiveness of the proposed method of representing and generating images in a binary latent space.

Social impacts. As a framework for representing and generating images, the proposed binary latent diffusion model shares with other generative models the risk of malicious uses such as deepfake. And the generated sample may exhibit bias residing in the training datasets.

References


[16] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In \textit{CVPR}, 2021. 3, 6, 7, 8, 1, 2


[34] Zhifeng Kong, Wei Ping, Jiayi Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. \textit{ICLR}, 2021. 2


[40] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. \textit{NeurIPS}, 2020. 1


[47] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2, 3, 6, 7, 8


[56] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021. 2

[57] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 1


[67] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *ICLR*, 2022. 7


[70] Yang Zhao, Jianwen Xie, and Ping Li. Learning energy-based generative models via coarse-to-fine expanding and sampling. In *ICLR*, 2020. 1
A. Noise Scheduler

We show in Figure A that with different noise schedulers, how 1 (blue curves) and 0 (red curves) in the binary latent representations progressively reach 0.5 in the diffusion processes. We use the Linear noise scheduler in all the experiments.

![Figure A. Noise schedulers. FID numbers are obtained with the 256 × 256 LSUN Churches experiments.](image)

B. Implementation Details

We use Adam as the default optimizer for all experiments. For the training of the binary auto-encoder in Section 3, we use a consistent learning rate of 5 × 10⁻⁴ with a linear learning rate warm-up for 10k iterations, and a batch size of 8. For C in (2), we use mean squared error, perception loss, and adversarial loss, with an equal \( \omega_i = 1.0 \) for each term.

For the training of the binary latent diffusion models in Section 4, we use a consistent learning rate of 1 × 10⁻⁴ with a linear learning rate warm-up for 10k iterations. All models are trained for 20K iterations. We do not use any additional regularizations such as drop-out or weight decay. We summarize the general training and sampling of the proposed binary diffusion models in Algorithm 1 and Algorithm 2, respectively.

All 256 × 256 unconditional binary latent diffusion models are trained with a batch size of 32. All 1024 × 1024 unconditional binary latent diffusion models are trained with a batch size of 16. All 256 × 256 conditional binary latent diffusion models are trained with a batch size of 256.

All training is conducted on cloud servers with V100 GPUs. All the speeding testing is conducted on a single RTX3090 GPU.

B.1. Network Architectures

For the binary auto-encoder in Section 3, we adopt a standard architecture using convolutional layers enhanced with self-attention layers, which is nearly identical to the one used in [16]. The only major difference is that the vector quantization layer is replaced by our binary latent layer implementing (1). This architecture gives a downsampling ratio of 16×. For the 1024 × 1024 high-resolution image generation experiments, of which the downsampling ratio raises to 32×, we simply insert one more downsampling block and one more upsampling block into the architecture.

For the conditional image generation experiments with ImageNet-1K, we train transformer networks with 24 layers and 768 feature channels with 12 heads in the self-attention layers. For the unconditional image generation experiments, we train transformer networks with 24 layers and 512 feature channels with 8 heads in the self-attention layers. In both conditional and unconditional image generation experiments, we use an extra time embedding token to specify the time step \( t \). And the class labels in conditional image generation with ImageNet are also specified by an additional token. Both class tokens and time-step tokens are trainable. We initialize the trainable position embedding in the transformer networks with 2D sine embedding.

B.2. Evaluation Metrics

FID. Fréchet Inception Distance (FID) is a commonly used metric for evaluating the quality of the generated images. It estimates the distance between two image distributions by comparing the mean and standard deviation of the deep image features extracted by a trained Inception network [57]. We conduct comparisons by generating 50,000 samples and compare them with the corresponding training dataset in every experiment.

PR. To comprehensively evaluate both the quality and mode-coverage of the generated samples compared to the training datasets, we further include precision-recall [35] as additional performance measurements.
Algorithm 1 Training procedure. We assume unconditional image generation with a batch size of one for the sake of discussion. The described training process can be easily extended to practical cases with arbitrary batch sizes by batching multiple samples.

1: Given: Trained encoder $\Psi$; Binary diffusion model $f_\theta$ parametrized by $T_\theta$; An image dataset $\mathbf{X}$.
2: Given: Diffusion steps $T$; Noise scheduler defined by $\{k_t\}_{t=1}^{T}$ and $\{b_t\}_{t=1}^{T}$; Training steps $I$; and $\lambda$ in (13).
3: Initializing $T_\theta$.
4: for Step $i = 1 : I$ do
5: Sampling image $\mathbf{x} \sim \mathbf{X}$, and time step $t \sim \{1, \ldots, T\}$.
6: Obtaining binary code $\mathbf{z}^i = \text{Bernoulli}(\sigma(\Psi(\mathbf{x})))$.
7: Obtaining $\mathbf{z}^i$ using $\mathbf{z}^i, t$, and noise scheduler with (5).
8: Predicting flipping probability $f_\theta(\mathbf{z}^i, t)$.
9: Obtaining predicted $\mathbf{z}^0$ as $p_\theta(\mathbf{z}^0) = (1 - \mathbf{z}^i) \odot f_\theta(\mathbf{z}^i, t) + \mathbf{z}^i \odot (1 - f_\theta(\mathbf{z}^i, t))$.
10: Obtaining predicted $p_\theta(\mathbf{z}^{i-1})$ using $p_\theta(\mathbf{z}^i)$ and $\mathbf{z}^i$ with (8).
11: Calculating loss $\mathcal{L}$ using (13).
12: Backpropagating $\mathcal{L}$ and updating $\theta$.
13: end for
14: Return Binary diffusion model $f_\theta$.

Algorithm 2 Sampling procedure. We assume unconditional image generation with a batch size of one for the sake of discussion.

1: Given: Trained decoder $\Phi$; Trained binary diffusion model $f_\theta$.
2: Given: Diffusion steps $T$; Noise scheduler defined by $\{k_t\}_{t=1}^{T}$ and $\{b_t\}_{t=1}^{T}$; Temperature $\tau$; Latent dimension specified by $h', w', c$, e.g., $h' = w' = 16, c = 32$ for the $256 \times 256$ image generation experiments.
3: Sampling $\mathbf{z}^i \sim \text{Bernoulli}(\mathbf{z}_{\text{init}})$, where $\mathbf{z}_{\text{init}} \in \mathbb{R}^{h' \times w' \times c}$ and contains 0.5 only.
4: for Step $t = T : 2$ do
5: Predicting $p_\theta(\mathbf{z}^{i-1})$ with $f_\theta(\mathbf{z}^i, t) = \sigma(T_\theta(\mathbf{z}^i, t)/\tau)$ and (8).
6: Sampling $\mathbf{z}^{i-1} \sim \text{Bernoulli}(p_\theta(\mathbf{z}^{i-1}))$
7: end for
8: Return the sampled image as $\Phi(\mathbf{z}^{i-1})$.

C. Additional Qualitative Results

C.1. Unconditional Image Generation

We present additional qualitative results of $256 \times 256$ LSUN Bedrooms and LSUN Churches unconditional image generation experiments in Figure B and Figure C, and qualitative comparisons of FFHQ unconditional image generation against Absorbing Diffusion models [6] and Latent Diffusion Models [47] in Figure D.

C.2. High-resolution Image Generation

We present additional qualitative results of $1024 \times 1024$ unconditional image generation experiments in Figure E and Figure F.

C.3. Conditional Image Generation

We present additional qualitative results of class-conditional image generation experiments and comparisons with BigGAN-deep [7], VQ-VAE-2 [44], VQGAN [16] and MaskGIT [8] in Figure I and Figure L.
C.4. Image Inpainting

We present additional image inpainting results with different mask patterns in Figure N.

C.5. Nearest Neighbours

To further show that our model is generating novel samples instead of overfitting to the training datasets, we compare the generated images with the corresponding training datasets using LPIPS, and visualize the top-10 nearest neighbours in Figure O.
Figure B. Additional unconditional image generation results and comparisons at 256 × 256 with the LSUN Churches dataset.
Figure C. Additional unconditional image generation results and comparisons at 256 × 256 with the LSUN Bedrooms dataset.
Figure D. Additional unconditional image generation results and comparisons at $256 \times 256$ with the FFHQ dataset.
Figure E. Additional high-resolution image generation results at $1024 \times 1024$ with the FFHQ dataset ($\tau = 0.8$).
Figure F. Additional high-resolution image generation results at $1024 \times 1024$ with the CelebA-HQ dataset ($\tau = 0.8$).
Figure I. Qualitative comparisons on class-conditional image generation with ImageNet class IDs: 22, 108, and 11.
Figure L. Qualitative comparisons on class-conditional image generation with ImageNet class IDs: 141, 154, and 1.
Figure M. Label-conditioned image generation with different scales of classifier-free guidance ($\omega$). Larger $\omega$ improves the sample quality at the cost of lower diversity.
Figure N. Conditional image inpainting with different masking patterns.
Figure O. Top-10 nearest neighbours in the training datasets of our generated samples. Results show that our model is not overfitting to the training datasets.