From Easy to Hard++: Promoting Differentially Private Image Synthesis Through Spatial-Frequency Curriculum

Chen Gong[†]
University of Virginia

Kecen Li[†]
University of Virginia

Zinan Lin
Microsoft Research

Tianhao Wang
University of Virginia

Abstract

To improve the quality of Differentially private (DP) synthetic images, most studies have focused on improving the core optimization techniques (e.g., DP-SGD). Recently, we have witnessed a paradigm shift that takes these techniques off the shelf and studies how to use them together to achieve the best results. One notable work is DP-FETA, which proposes using 'central images' for 'warming up' the DP training and then using traditional DP-SGD.

Inspired by DP-FETA, we are curious whether there are other such tools we can use together with DP-SGD. We first observe that using 'central images' mainly works for datasets where there are many samples that look similar. To handle scenarios where images could vary significantly, we propose FETA-Pro, which introduces *frequency features* as 'training shortcuts.' The complexity of frequency features lies between that of spatial features (captured by 'central images') and full images, allowing for a finer-grained curriculum for DP training. To incorporate these two types of shortcuts together, one challenge is to handle the training discrepancy between spatial and frequency features. To address it, we leverage the pipeline generation property of generative models (instead of having one model trained with multiple features/objectives, we can have multiple models working on different features, then feed the generated results from one model into another) and use a more flexible design. Specifically, FETA-Pro introduces an *auxiliary generator* to produce images aligned with noisy frequency features. Then, another model is trained with these images, together with spatial features and DP-SGD. Evaluated across five sensitive image datasets, FETA-Pro shows an average of 25.7% higher fidelity and 4.1% greater utility than the best-performing baseline, under a privacy budget $\varepsilon = 1$.

1 Introduction

Privacy-preserving image synthesis aims to create artificial image data that retains the characteristics of real images, facilitating image data sharing within and between organizations while reducing privacy concerns [1–3]. Differentially private (DP) image synthesis [1, 4–8] provides a rigorous theoretical framework to quantify and limit privacy leakage from real image data using synthetic datasets.

Most DP image synthesis methods rely on non-sensitive public resources released on open-source platforms like HuggingFace. They include pre-training synthesizers on public datasets [5, 9, 10] or leveraging pre-trained models [6, 7] to enhance synthetic performance. Tramèr et al. [11] note that public datasets may potentially compromise privacy. For example, the GPT-2 language model, pretrained on public web data, memorized the phone number of an individual named 'Peter W.' [12, 13]. In addition, in reality, we may not find public datasets that are aligned well with sensitive datasets, and in such cases, these methods perform badly [4]. Therefore, *we aim at developing an effective DP image synthesis method that does not rely on public resources.*

Existing Methods. Various works focus on developing DP Stochastic Gradient Descent (DP-SGD) [14] to improve DP image synthesis [9, 10]. Recently, we witnessed a paradigm shift – DP-FETA [15] that uses 'central images' to warm up synthesizers. The 'central images' can be regarded as a 'shortcut' to improve the training efficiency of DP-SGD [15]. Specifically, DP-FETA (1) extracts 'central images' by averaging pixel values of aggregated images to warm up synthesizers, and (2) then fine-tunes synthesizers on original images using DP-SGD. DP-FETA achieves state-of-the-art performance in DP image synthesis without using public resources. However, 'central images' capture only simplistic characteristics, making them difficult to handle complex images and most effective for datasets where many samples are similar.

Our proposal. We are curious whether there are other such 'training shortcuts' we can use together with DP-SGD, and refine them to enhance DP-SGD for DP image synthesis. To address the issue of the limited capacity of 'central images,' we introduce frequency information as 'training shortcuts.' This is inspired by previous works [16, 17] that highlight the effectiveness of frequency features, and we use them as a

[†]Equal contribution. Kecen works as a remote intern at UVA.

complement to spatial features. As detailed in Section 3.1, frequency features provide richer information, like global structures and textures, about sensitive datasets. Overall, we propose FETA-Pro, a spatial-frequency curriculum, which integrates frequency and spatial 'training shortcut.'

The next question is when to use the frequency training shortcut. Wang et al. [18] suggest that the transition from easy to hard curriculum can benefit the completion of target tasks. As introduced in Section 3.1, the complexity of frequency features falls between that of spatial features (captured by 'central images,) and the full image, allowing for a more fine-grained curriculum for DP training. Thus, FETA-Pro learns from the order of 'spatial features $\rightarrow$ frequency features $\rightarrow$ images.' Referring to DP-FETA [15], we use 'central images' as spatial features, and extract frequency features using the Fourier Transform [19]. We list challenges and solutions when incorporating frequency features as a 'training shortcut.'

- *Challenge I.* A key challenge arises in the training discrepancy between the spatial and frequency features. To address this challenge, we propose learning spatial and frequency features on a unified representation. Specifically, FETA-Pro first extracts frequency features and introduces noise to ensure DP. Subsequently, FETA-Pro introduces an *auxiliary generator* to learn to generate images aligned with the noisy frequency features. These synthetic images implicitly encode the frequency features, which are used to learn the underlying frequency features.

- *Challenge II.* For the design of auxiliary generators, diffusion models, which are used in FETA-Pro as synthesizers, referring to previous works [5, 15], are not suitable as auxiliary generators. Diffusion models learn via a forward diffusion process, incrementally introducing noise to a clean input [20]. Determining the precise frequency features at each step of the diffusion process is challenging. Therefore, we use *one-step* generators in the generative adversarial network (GAN) [21], which generates images only via one step, as the auxiliary generator.

Thus, instead of training one model on multiple features or objectives, FETA-Pro leverages the pipeline generation property of generative models. This allows us to use a more flexible design by having multiple models work on different features. We discuss more about the challenges and solutions in Section 3.3.1. Finally, the warm-up diffusion model is trained on the original sensitive images leveraging DP-SGD [14]. Figure 1 presents the framework of FETA-Pro.

Evaluations. We compare FETA-Pro to seven baselines on five widely used sensitive datasets. Compared to the state-of-the-art method DP-FETA [15], FETA-Pro achieves 25.7% higher fidelity and 4.1% greater utility for synthetic images, under privacy budget $\varepsilon = 1$. In particular, on the human face dataset `CelebA` [22] and the medical dataset `Camelyon`, FETA-Pro enhances synthetic image quality by 20.2% and 41.2% in FID [23], and improves downstream classification task accuracy (Acc) by 7.7% and 6.7%, respectively, compared to DP-FETA [15]. Besides, the synthetic performance of FETA-Pro converges more rapidly than that of baselines. These results show that FETA-Pro incorporates frequency features as the shortcut, significantly improving the performance of DP image synthesis. Additionally, we study various privacy budget allocation strategies for spatial and frequency features in FETA-Pro. We observe that allocation strategies highly affect synthetic image quality. Overall, we recommend allocating privacy budget cost ratios ordered as the complexity of features, like 'spatial features < frequency features < images.'

FETA-Pro also maintains good efficiency: Compared to DP-FETA [15], for `CIFAR-10`, FETA-Pro incurs only a 0.3% increase in training (approximately 0.06 hours).

Table 1: The type of synthesizer used in existing methods. The first six methods listed above the horizontal line rely solely on sensitive data, and the other methods use public resources.

Method	Training Methods	Type of Synthesizer
DP-MERF [19]	Regular SGD	GAN
DP-NTK [24]	Regular SGD	GAN
DP-Kernel [25]	Regular SGD	GAN
DP-GAN [26]	DP-SGD	GAN
DPDM [27]	DP-SGD	Diffusion Model
DP-FETA [15]	DP-SGD + Shortcut	Diffusion Model
FETA-Pro (Ours)	DP-SGD + Shortcut	GAN & Diffusion Model
PDP-Diffusion [28]	DP-SGD	Diffusion Model
PE [6]	-	APIs
PrivImage [5]	DP-SGD	Diffusion Model
DP-LDM [9]	DP-SGD	Diffusion Model
DP-LoRA [10]	DP-SGD	Diffusion Model

Contributions. We list our contributions as follows,

- This paper proposes FETA-Pro, a spatial-frequency curriculum, that refines 'training shortcuts' by incorporating spatial and frequency features to warm up synthesizers.

- Table 1 shows that current methods predominantly use a single type of model as the synthesizer. FETA-Pro redefines DP image synthesis by integrating multiple synthesizers' strengths (i.e., diffusion models and GANs), surpassing traditional single-synthesizer approaches to enhance performance and paving a path for further methods.

- Comprehensive experiments present that FETA-Pro accelerates DP diffusion model training while achieving state-of-the-art fidelity and utility across five image datasets, and only incurs minimal additional computer resources.

2 Backgrounds

2.1 Differential Privacy

Differential privacy (DP) [29] quantifies the risk of individual privacy leakage within a dataset during data processing,

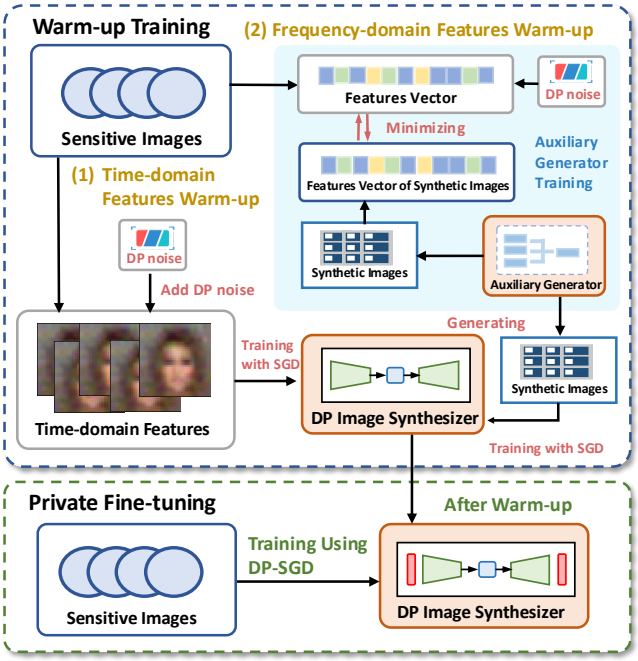


Figure 1: The framework of FETA-Pro. During warm-up, FETA-Pro extracts spatial features to train the synthesizer. Then, FETA-Pro introduces auxiliary generators to generate images aligning the noisy frequency features, and training synthesizers on these synthetic images. Then, we train the warmed-up synthesizers on sensitive images using DP-SGD.

serving as the gold standard for privacy preservation. We introduce the concept of DP as follows.

Definition 1 (*Differential Privacy* [29])**.** *A randomized mechanism $\mathcal{Q}$ satisfies (ε,δ)-DP if for any two neighboring datasets D and D', the following condition holds,*

$$\Pr[\mathcal{Q}(D)\in\mathcal{O}] \le e^{\varepsilon}\Pr[\mathcal{Q}(D')\in\mathcal{O}]+\delta. \tag{1}$$

Here, $\mathcal{O}$ represents a set of possible outputs of the algorithm $\mathcal{Q}$. $\mathcal{Q}$ is also referred to as the query function in the literature [2]. The parameters (ε,δ) quantify the privacy loss, with both being non-negative. A smaller ε indicates stronger privacy protection, while δ represents the probability of failure, where a smaller δ reduces the likelihood that the privacy guarantees provided by ε are violated [29]. Datasets D and D' are considered neighboring if one can be obtained from the other by adding or removing a single image.

If an algorithm $\mathcal{Q}$ satisfies (ε,δ)-DP, any post-processing function $\mathcal{F}$, applied as $\mathcal{F}\circ\mathcal{Q}$ and $\mathcal{F}$ depending only on $\mathcal{Q}$'s output, incurs no additional privacy loss [29].

DP-SGD. In machine learning, a widely adopted method for achieving DP is DP-SGD [14]. This approach modifies standard SGD by computing gradients using Poisson-sampled mini-batches, clipping the gradients of the model's parameters, and adding Gaussian noise to the clipped gradients during training. Formally, we sample a sub-batch of images $D_s^{\text{sub}} = \{h_i\}_{i=1}^{B}$ from the sensitive dataset D_s with sampling probability q. Drawing on the implementation of DP-SGD [14], we use Poisson sampling at each iteration, where the batch size B is not fixed but follows an expected batch size $B^* = qN$. N means the size of the sensitive dataset. The parameters θ of the synthesizer are updated via the following noisy gradient,

$$\eta\left(\frac{1}{B^*}\sum_{i=1}^{B}\text{Clip}\left(\nabla\mathcal{L}(\theta,h_i),C\right)+\frac{C}{B^*}\mathcal{N}(0,\sigma^2\mathbb{I})\right), \tag{2}$$

where $\mathcal{L}$ is the objective function of diffusion models, and η is the learning rate and σ^2 is the variance of Gaussian noise. $\text{Clip}(\nabla\mathcal{L},C) = \min\left\{1,\frac{C}{||\nabla\mathcal{L}||_2}\right\}\nabla\mathcal{L}$ clips the norm of gradient smaller than the hyper-parameter C.

This paper uses Rényi DP (RDP) [30] to track the privacy loss, as detailed in Appendix A.

DP Image Synthesis. The goal is to generate synthetic datasets that closely resemble real data while preserving the privacy of individual data points. This allows organizations to share synthetic images for various applications with reduced privacy concerns.

Table 1 lists synthesizers used in existing methods. Prior work primarily focuses on using a single synthesizer type. Given the exceptional generative capabilities of diffusion models, recent methods increasingly emphasize their adoption [5, 9, 10], while FETA-Pro combines the strengths of multiple synthesizers to enhance synthetic performance. Section 7 presents more details of current methods.

2.2 Diffusion Model

Diffusion models, as synthesizers, achieve the SOTA performance and are widely adopted in DP image synthesis [5–7, 15, 27, 28]. This work builds upon prior studies and uses diffusion models as synthesizers. Diffusion models [20] operate through two key processes:

- The *forward diffusion process*, which gradually adds noise to a clean image h_0, generating a sequence of increasingly noisy images $\{h_t\}_{t=1}^{T}$ until it approximates pure random noise, where T denotes the total number of noising steps.
- The *reverse diffusion process*, which progressively denoises random noise to reconstruct a clean image.

In the forward diffusion process, the transition between consecutive noisy images, denoted $p(h_t|h_{t-1})$, follows a multidimensional Gaussian distribution, $p(h_t|h_{t-1}) = \mathcal{N}\left(h_t;\sqrt{1-\beta_t}h_{t-1},\beta_t\mathbb{I}\right)$, where β_t is a hyper-parameter controlling the noise variance at each step. We define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. Consequently, the likelihood of a

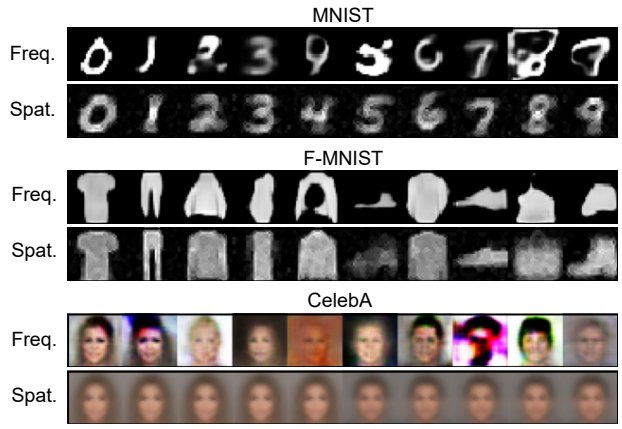


Figure 2: Sampled images generated leveraging frequency ('Freq.') and spatial ('Spat.') domain features under $\varepsilon = 1$.

noisy image x_t given the clean image h_0 is, $p(h_t|h_0) = \mathcal{N}\left(h_t; \sqrt{\bar{\alpha}_t}h_0, (1-\bar{\alpha}_t)\mathbb{I}\right)$. This enables direct sampling of h_t from h_0 in closed form, $h_t = \sqrt{\bar{\alpha}_t}h_0 + e_t\sqrt{1-\bar{\alpha}_t}, e_t \sim \mathcal{N}(0,\mathbb{I})$. The objective of diffusion models is to train a neural network to predict the noise added at each step [20]:

$$\mathcal{L} = \mathbb{E}_{h_0\sim D, t\sim U\{1,T\}, e_t\sim\mathcal{N}(0,\mathbb{I})} \|e_t - e_\theta(h_t,t)\|_2^2 \qquad (3)$$

where D is the dataset of clean images, and $e_\theta(h_t,t)$ is a denoising network parameterized by θ. $U\{1,T\}$ denotes a discrete uniform distribution over the integers from 1 to T. This network learns to predict the noise ε_t in a noisy image h_t at step t. Once trained, ε_θ enables the generation of clean images by denoising random Gaussian noise.

2.3 Frequency Domain Features of Images

Fourier Transform (FT) is a tool for analyzing images in the frequency domain. For a continuous image denoted as a function $I(x,y)$ over a spatial domain $[0,l]\times[0,w]$, where l and w represent height and width, the FT transforms the image into frequency components $F(u,v)$,

$$F(u,v) = \int_0^g \int_0^w I(x,y)e^{-j2\pi(ux+vy)}\,dx\,dy, \qquad (4)$$

where u,v are continuous frequency variables representing spatial frequencies in horizontal and vertical directions [31].

2.4 Training Shortcuts

DP-FETA introduces using 'central image' to augment DP-SGD, which we view as a shortcut for training. In this paper, we provide a formal description of this concept.

'Training shortcuts' are simpler representations of a sensitive dataset, D_s, that (1) are beneficial for the synthesizer to learn complex images and (2) only need minimal privacy loss to obtain, than the original, complex images. This knowledge is formally expressed as $F(D_s)$, where $F(\cdot)$ denotes the knowledge extraction method. Then, the synthesizer, e, is trained with DP to learn the mapping from simpler knowledge to the original data, following the order of '$e \to F(D_s) \to D_s$.'

Table 2: The entropy and texture complexity of images generated leveraging frequency and spatial features. Higher values indicate greater image complexity. 'Text. Comp.' is the abbreviation for texture complexity.

Feature	CIFAR-10		CelebA	
	Entropy	Text. Comp.	Entropy	Text. Comp.
Spatial	0.522	1.228	0.857	1.857
Frequency	1.064	1.560	1.192	1.966
Original Images	1.243	2.169	1.336	2.363

3 Methodology

This section details the spatial-frequency curriculum, FETA-Pro, including the motivation and technical details.

3.1 Motivation

DP-FETA uses 'central images' (serve as basic spatial features) extracted from sensitive data as a training shortcut. However, these features capture only coarse and simplistic characteristics, making it difficult to handle complex datasets. For example, as shown in Table 4, the FID on the `Camelyon` dataset is only 52.8 under privacy budget $\varepsilon = 1$.

Previous works [16,17] have shown that frequency features, which capture global structures and textures, are effective for various tasks. This inspires us to integrate frequency features as a training shortcut from the two perspectives.

- *Complementing spatial features.* Figure 2 displays synthetic images generated by synthesizers that respectively use frequency and spatial features under $\varepsilon = 1$. This figure shows that spatial features capture the shape and color details of images, while frequency features emphasize texture characteristics and more clearly capture edge variations.

- *Fine-grained curriculum.* Wang et al. [18] advocate for fine-grained curricula to enhance task effectiveness. As presented in Table 2, we observe that frequency features possess a complexity between spatial features and full images. We provide more details of 'entropy' and 'complexity' in Appendix C.3. This observation forms the basis for our curriculum design in DP image synthesis. FETA-Pro thus orders features by complexity, progressing from 'spatial features' to 'frequency features' and finally to 'images.'

Section 5.2 presents that integrating these two types of features achieves better synthetic performance than using either one in isolation. Besides, Section 5.2 validates that

Algorithm 1: Feature Extractions in FETA-Pro.

Input : Sensitive dataset D_s with estimated size N^*; number of central images N_t, batch size B_t; pre-defined vector length K; the height and width of images l and w.

// Spatial feature extraction

Init noisy central image dataset $D_s^t = \varnothing$;

while $len(D_s^t) < N_t$ **do**

  Sample subset $D_s^{\text{sub}} = \{h_i\}_{i=1}^{B_t}$ from D_s;

  Calculate noisy central image $\tilde{h}$ using Equation (7);

  $D_s^t = D_s^t \cup \{\tilde{h}\}$;

end

// Frequency feature extraction

for $h_i \in D_s$ **do**

  Vectorize $h_i \in \mathbb{R}^{l \times w}$ to $h_i \in \mathbb{R}^d$, where $d = l \times w$;

  for $j = 1, 2, \cdots, K/2$ **do**

    $\omega_j \in \mathbb{R}^d \sim \mathcal{N}(0, \mathbb{I})$;

    $\phi_j(h_i) = \sqrt{2/K}\cos(\omega_j^\top h_i)$;

    $\phi_{j+K/2}(h_i) = \sqrt{2/K}\sin(\omega_j^\top h_i)$;

  end

  $\phi(h_i) = (\phi_1(h_i), \ldots, \phi_K(h_i))^\top$,

end

Calculate mean frequency features: $\mu = \frac{1}{N^*}\sum_{i=1}^{N}\phi(h_i), \mu \in \mathbb{R}^K$;

Add noise to μ for DP using Equation (6) and obtain $\tilde{\mu}$;

Output : Spatial and frequency feature, D_s^t and $\tilde{\mu}$.

FETA-Pro achieves the best synthetic performance compared to learn spatial and frequency features concurrently and learn through prioritizing frequency features over spatial features.

3.2 Feature Extraction

This section explains how to extract frequency and spatial features from sensitive datasets under DP as training shortcuts.

Frequency Features Shortcuts. Random Fourier Features [19, 32, 33] are preferred over the traditional Fourier Transform for image feature extraction in image processing due to their computational efficiency and ability to represent arbitrary dimensions of images as fixed-length vectors [32]. We refer to implementations in previous works [19]. Given an image h, its random Fourier features are given by, $\phi(h) = (\phi_1(h), \ldots, \phi_K(h))^\top$, where K is the pre-defined feature dimension, and each coordinate is calculated by,

$$\begin{cases} \phi_j(h) = \sqrt{2/K}\cos(\omega_j^\top h) \\ \phi_{j+K/2}(h) = \sqrt{2/K}\sin(\omega_j^\top h) \end{cases}, \tag{5}$$

where $j = \{1, \ldots, K/2\}$, and $\omega_j \in \mathbb{R}^d$ is a random frequency vector, with each component independently drawn from a standard normal distribution. The $d = l \times w$ is the dimension of the vectorized image h. l and w represent the height and width of images. The term $\omega_j^\top h$ is the inner product $\langle \omega_j, h \rangle$, projecting the image onto the random frequency ω_j, enabling the construction of cosine and sine features to capture frequency features [31, 32].

For a sensitive image dataset $D_s = \{h_i\}_{i=1}^N$ with N images, the mean frequency features are, $\mu = \frac{1}{N^*}\sum_{i=1}^{N}\phi(h_i)$, where N^* approximates dataset size N as introduced in Appendix B.

$$\tilde{\mu} = \mu + \mathcal{N}(0, \sigma_f^2 \Delta_f^2 \mathbb{I}), \tag{6}$$

where σ_f^2 is a hyper-parameter describing the scale of Gaussian noise, and the global sensitivity $\Delta_f = 1/N^*$. Theorem 1 proves that this process ensures DP.

Theorem 1. *The query of frequency features of D_s has global sensitivity $\Delta_f = 1/N^*$. For any $\alpha > 1$, incorporating noise $\mathcal{N}\left(0, \sigma_f^2 \Delta_f^2 \mathbb{I}\right)$ into the mean random Fourier feature μ makes the query results satisfy (α, γ)-RDP for some γ.*

We detail (α, γ)-RDP in Appendix A. Besides, we provide proof of Theorem 1 in Appendix B.

Spatial Features Shortcuts. We borrow the approach of extracting spatial features from DP-FETA [15], which uses central images as spatial shortcuts. We first sample a subset of B_t sensitive images, $D_s^{\text{sub}} = \{h_i\}_{i=1}^{B_t}$, from the sensitive dataset $D_s = \{h_i\}_{i=1}^N$, consisting of N images, using Poisson subsampling with probability q_t. The batch size B_t varies and is not fixed in each step. We only have access to the expected batch size, $B_t^* = q_t N^*$, where N^* is the estimated dataset size. Each sensitive image is then clipped to ensure a bounded L_2 norm, defined as $h_i^c = \min\left\{1, \frac{C_t}{\|h_i\|_2}\right\} \cdot h_i$, where C_t is a hyper-parameter, guaranteeing $\|h_i^c\|_2 \leq C_t$. The central image is computed as $h^{\text{spat}} = \frac{1}{B_t}\sum_{i=1}^{B_t^*} h_i^c$. Then, Gaussian noise is added to the central images to obtain final spatial features $\tilde{h}$,

$$\frac{1}{B_t^*}\sum_{i=1}^{B_t} \min\left\{1, \frac{C_t}{||h_i||_2}\right\} \cdot h_i + \mathcal{N}\left(0, \sigma_t^2 \Delta_{\text{spat}}^2 \mathbb{I}\right), \tag{7}$$

where σ_t^2 is a hyper-parameter that controls the scale of the noise, and $\Delta_{\text{spat}} = C_t / B_t^*$. We can prove that this ensures DP.

Theorem 2. *The query of spatial features h^{spat} has global sensitivity $\Delta_{spat} = C_t/B_t^*$. For any $\alpha > 1$, incorporating noise $\mathcal{N}\left(0, \sigma_t^2 \Delta_{spat}^2 \mathbb{I}\right)$ into the spatial features h^{spat} makes the query results satisfy (α, γ)-RDP for some γ.*

We provide the proof of Theorem 2 in Appendix B. Repeating the above process N_t times, we can obtain the noisy central image dataset $D_s^t = \{\tilde{h}_i\}_{i=1}^{N_t}$ as the spatial features. FETA-Pro uses RDP composition to analyze the privacy cost of extracted spatial and frequency features. We elaborate on the privacy analysis in Section 3.4.

Algorithm 1 presents the workflow of extracting spatial and frequency features from sensitive datasets under DP.

Algorithm 2: The workflow of FETA-Pro.

Input : Diffusion model e_θ parameterized with θ; sensitive dataset D_s; spatial and frequency feature, D_s^t and $\tilde{\mu}$; auxiliary generator $G_{\theta'}$ parameterized with θ'; training epoch of auxiliary generator M; training batch size of auxiliary generator B_f; the number of images sampled from the auxiliary generator N_f.

// Spatial domain warm-up

Train the synthesizer e_θ on D_s^t using Equation (3);

// Frequency domain warm-up

Init $m = 0$;

while $m \leq M$ **do**

  Leverage $G_{\theta'}$ to generate images $D^{\text{tmp}} = \{h_i\}_{i=1}^{B_f}$;

  Extracting frequency features $\mu^s(G_{\theta'})$ from D^{tmp} using Line 7-Line 16 in Algorithm 1;

  Update θ' by minimizing: $\mathcal{L}(\theta') = \|\tilde{\mu} - \mu^s(G_{\theta'})\|_2$;

  $m = m + 1$;

end

Leverage trained $G_{\theta'}$ to generate images $D_s^f = \{h_i\}_{i=1}^{N_f}$;

Train the synthesizer e_θ on D_s^f using Equation (3);

// Private Fine-tuning

Fine-tuning the synthesizer e_θ on D_s using DP-SGD;

Output : Well-trained DP image synthesizer e_θ.

Features Extraction for Labeled Data. When data labels are available–a common scenario in image generation tasks–we can partition the sensitive dataset D_s into multiple disjoint subsets based on labels. Specifically, we categorize D_s into C disjoint subsets, $D_s = [D_s^1, D_s^2, \cdots, D_s^C]$, based on their labels, where C is the number of categories. Then, we extract spatial and frequency features for each subset. By the parallel composition property [29], processing disjoint subsets incurs no additional privacy cost, yielding the same privacy guarantees as if the entire dataset were processed without partitioning.

For frequency features extraction, we first calculate the random Fourier features of each image and aggregate them in each subset $[D_s^1, D_s^2, \cdots, D_s^C]$. Then, we obtain the mean frequency features, $\mu = [\mu_1, \mu_2, \cdots, \mu_C]$, where $\mu_i = \frac{1}{m}\sum_{h \in D_s^i} \phi(h)$ and $\mu \in \mathbb{R}^{K \cdot C}$. K is the pre-defined feature dimension. Due to the disjoint nature of the subsets, the privacy guarantee remains the same as that in Theorem 1.

Theorem 3. *Consider partitioning the dataset as $D_s = [D_s^1, D_s^2, \cdots, D_s^C]$, where the subsets are disjoint. By applying Equation (6) to each subset, we generate noisy frequency features. The combined features have a dimension of $K \cdot C$, where K is the predefined feature dimension and C is the number of subsets. Because the subsets are disjoint, this mechanism has the same (α, γ)-RDP guarantee as that in Theorem 1.*

The proof of Theorem 3 is provided in Appendix B. We then obtain the target frequency features using Equation (6).

For spatial features, when sampling a batch of images to query a central image, we ensure that all sampled images are from a single subset. Similarly, the query of spatial features keeps consistent sensitivity as presented in Theorem 2.

3.3 Private Training

This section first introduces challenges in private training and then elaborates on technical details.

3.3.1 Challenges and Solutions

We present the challenges of incorporating frequency features as training shortcuts and solutions in FETA-Pro as follows.

- **The discrepancy between spatial and frequency features poses a challenge for their learning together.** Spatial and frequency features exist in different domains; spatial features capture localized patterns in the data's original representation, while frequency features, such as those from random Fourier transforms, represent global oscillatory characteristics in a transformed space. [34]. It is unclear how to train synthesizers on these two distinct domains together.

  To solve this challenge, we hope that synthesizers learn these features on a unified representation. Since diffusion models primarily learn in the spatial domain [20, 35], we convert frequency feature learning to this domain. Specifically, FETA-Pro first extracts frequency features from sensitive images under DP. An *auxiliary generator* is then trained to produce images that align with these noisy frequency features. We consider these synthetic images involving frequency features of sensitive images.

- **Using diffusion models (the model of synthesizers) as the auxiliary generator is not suitable.** Diffusion models learn via a forward diffusion process, incrementally introducing noise to clean inputs [20]. The specific frequency features of sensitive images at each step of the diffusion process remain unknown. We only have access to the frequency features of the clean sensitive images. Thus, multiple-step generation synthesizers (i.e., diffusion models) are suboptimal.

  We adopt the *one-step* generator [21], which generates images in a single step, as the auxiliary generator. The auxiliary generator learns to create images that approximate the noisy frequency features. These images, which incorporate frequency features, are then used by the main generator to warm up the diffusion model.

This design combines the pipeline generation property of different generative models, enabling different models to handle specific features and resulting in a more flexible framework for DP image synthesis. Table 5 of Section 5.1 presents that learning frequency features directly using synthesizers without auxiliary generators and using diffusion models as

auxiliary generators achieves inferior performance compared to our proposed FETA-Pro.

3.3.2 Technical Details

Warm Up. The spatial features consist of a noisy central image dataset D_s^t. We can first directly train the diffusion model on this dataset for warm-up.

As we discussed in Section 3.3.1, we introduce an auxiliary generator to learn to generate images aligning with the noisy frequency features $\tilde{\mu}$ from the sensitive images D_s. Then, synthetic images from the auxiliary generator, which involve the frequency features of sensitive datasets, are used to further warm up the diffusion models. Specifically, we employ the generator from a GAN, which produces images from a random Gaussian vector z in a single step, as the auxiliary generator. In each train epoch, we use a vector set $\{z_i\}_{i=1}^{B_f}$ to generate a synthetic image dataset $D^{\text{tmp}} = \{h_i\}_{i=1}^{B_f} = \{G_{\theta'}(z_i)\}_{i=1}^{B_f}$, where B_f refers to the size of synthetic images and θ' is the network parameter. Referring to Equation (5), we can obtain the frequency features of D^{tmp}, $\mu^s = \frac{1}{B_f}\sum_{i=1}^{B_f}\phi(h_i), h_i \in D^{\text{tmp}}$. We hope that the μ^s are close to the noisy features of sensitive datasets D_s, so the objective of the auxiliary generator is,

$$\mathcal{L}(\theta') = \|\tilde{\mu} - \mu^s(G_{\theta'})\|_2 \tag{8}$$

Note that we only borrow the generator in GAN. Unlike traditional GAN training, which relies on a critic to provide an adversarial loss to indirectly guide the generator [21, 36], our critic (Equation (8)) is explicitly defined. This makes the training more efficient and avoids the training instability issue in traditional GANs [19, 36, 37].

DP-SGD Fine-tuning. Then, we fine-tune the warmed-up diffusion model on the original sensitive images to learn more complex features of the images. To achieve DP, we add Gaussian noise to the clipped training gradients and use the noisy gradient to update the model parameters following standard DP-SGD [14], as introduced in Section 2.1. Algorithm 2 summarizes the workflow of FETA-Pro.

3.4 Privacy Analysis

In FETA-Pro, three processes consume the privacy budget: (1) querying the frequency features, (2) querying the central images for spatial features, and (3) fine-tuning the warm-up diffusion model on the sensitive dataset using DP-SGD. According to Renyi DP (RDP) [38], these three processes satisfy (α, γ_t)-RDP, (α, γ_f)-RDP and (α, γ_d)-RDP, respectively. Specifically, (α, γ_t) is determined by the noise scale σ_t, and we use the whole sensitive dataset. (α, γ_f) is determined by the number of central images N_t, sample ratio q_t and noise scale σ_t. (α, γ_d) is determined by the fine-tuning iteration t_d, sample ratio q_d and noise scale σ_d [14]. Each process of FETA-Pro can be viewed as a Sub-sampled Gaussian Mechanism (SGM) [30].

Table 3: The data split and number of categories of sensitive image datasets used in our experiments.

Sensitive Datasets	Training	Validation	Test	Category
MNIST	55,000	5,000	10,000	10
F-MNIST	55,000	5,000	10,000	10
CIFAR-10	45,000	5,000	10,000	10
CelebA	162,770	19,867	19,962	2
Camelyon	302,436	34,904	85,054	2

According to the RDP composition theorem [38], FETA-Pro satisfies $(\alpha, \gamma_t + \gamma_f + \gamma_d)$-RDP. We introduce the concept and property of RDP in Appendix A.

To make FETA-Pro satisfy a given (ε, δ)-DP, we determine privacy parameters following three steps: (1) We set the number of central images N_t, sample ratio q_t and noise scale σ_t, σ_f to obtain the RDP costs (α, γ_t) and (α, γ_f). (2) We fix the fine-tuning iterations t_d and sample ratio q_d, and then the RDP cost of DP-SGD is a function of noise scale σ_d as $(\alpha, \gamma_d(\sigma_d))$. (3) We try different σ_d to obtain the corresponding (ε, δ)-DP [38], until meeting the given privacy budget. Appendix D.2[1] [39] discusses the privacy calculation in FETA-Pro. Appendix D.3 [39] presents results under another privacy accounting method, Privacy Random Variable (PRV) [40].

4 Experimental Setup

Baselines. FETA-Pro achieves DP image synthesis without relying on public resources, such as public datasets or pre-trained models. Therefore, we select baselines under the same constraint, including DP-MERF [19], DP-NTK [24], DP-Kernel [25], GS-WGAN [41], DP-GAN [26], DPDM [27], and DP-FETA [15]. The implementations use the open-source DP image synthesis benchmark, DPImageBench [4].[2] We provide more baselines details in the Appendix C.1.

Implementations. All experiments are implemented with Python 3.8 on a server with 4 NVIDIA GeForce A6000 Ada and 512GB of memory. We aim at conditional generation for these datasets (i.e., each generated image is associated with the class label). Following practical adoption in DPImageBench [4], we set DP parameter $\delta = 1/(N_{\text{priv}} \times \log N_{\text{priv}})$, where N_{priv} means the number of samples in training sensitive datasets as presented in Table 3, and $\varepsilon = \{1, 10\}$.

Following DPImageBench, the synthesizer sizes are 5.6M parameters for GAN-based methods (DP-MERF, DP-NTK, DP-Kernel, GS-WGAN, DP-GAN) and 3.8M for diffusion-based methods (DPDM, DP-FETA, FETA-Pro). For GAN-based methods, the generator and discriminator sizes are 3.8M and 1.8M parameters, respectively. We recognize that larger synthesizer models may enhance synthetic performance. However, for a fair comparison, this paper adopts the synthesizer

[1] Due to space limitations, please refer to our full version [39] for additional appendices (Appendix C.3-F).

[2] https://github.com/2019ChenGong/DPImageBench

Table 4: FID and Acc (%) of FETA-Pro and seven baselines on MNIST, F-MNIST, CIFAR-10, CelebA and Camelyon with $\varepsilon = \{1, 10\}$. The best and second-best values are highlighted in bold and underlined in each column.

Method	$\varepsilon = 1$										$\varepsilon = 10$									
	MNIST		F-MNIST		CIFAR-10		CelebA		Camelyon		MNIST		F-MNIST		CIFAR-10		CelebA		Camelyon	
	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc	FID	Acc
DP-MERF	113.7	80.3	66.3	62.2	203.9	27.2	176.2	81.0	278.3	60.4	106.3	81.3	106.4	62.2	214.1	29.0	147.9	81.2	251.6	58.3
DP-NTK	382.1	50.0	253.1	64.4	413.0	17.0	350.4	61.2	335.5	53.1	69.2	91.3	120.5	76.3	346.9	28.2	227.8	64.2	234.5	64.1
DP-Kernel	33.7	94.0	63.4	68.4	184.8	26.4	140.3	83.0	254.1	68.0	38.9	93.6	74.2	70.0	161.4	25.1	128.8	83.7	217.3	68.7
GS-WGAN	48.8	72.4	99.4	52.7	259.7	20.4	611.8	61.4	421.3	52.1	47.7	75.3	97.2	56.7	194.4	21.3	290.0	61.5	291.8	58.9
DP-GAN	57.0	92.4	74.8	71.8	187.5	26.2	112.5	77.9	132.2	83.2	30.3	92.7	76.9	70.3	138.7	30.5	31.7	89.2	66.9	79.6
DPDM	36.1	89.2	28.8	76.4	206.4	28.9	153.9	74.5	111.9	80.6	4.4	97.7	17.1	85.6	110.1	36.8	28.8	91.8	29.2	79.5
DP-FETA	13.7	95.6	31.4	81.7	139.8	35.1	60.2	82.3	52.8	77.3	3.4	98.1	13.3	87.3	95.3	43.3	24.8	94.2	27.8	82.9
FETA-Pro	**8.0**	**97.3**	**27.8**	**83.4**	**120.4**	**37.9**	**48.0**	**90.0**	**31.0**	**84.0**	**2.5**	**98.6**	**11.6**	**88.1**	**69.0**	**47.0**	**20.0**	**95.2**	**20.3**	**84.3**

size used in prior work. We provide more details of hyper-parameter settings in Appendix C.4 [39].

Investigated Tasks and Datasets. We perform experiments on five image datasets MNIST [42], FashionMNIST [43] (F-MNIST), CIFAR-10 [44], CelebA [22], and Camelyon [45]. The investigated datasets are prevalently used in previous DP image synthesis methods [15, 28]. We provide more details of the investigated datasets in Appendix C.2.

Evaluation Metrics. We assess the fidelity and utility of synthetic datasets using two widely adopted metrics [4, 6, 7, 15, 27]: Fréchet Inception Distance (FID) and downstream classification accuracy. Our implementation follows the practical framework outlined in DPImageBench [4]. We generate 60,000 synthetic images for evaluations. Please refer to Appendix C.3 for more details.

5 EMPIRICAL EVALUATIONS

This section studies the effectiveness of FETA-Pro by answering three Research Questions (RQs) as follows. **(RQ1)** Does FETA-Pro outperform the seven baseline methods across the studied image datasets? **(RQ2)** How do the frequency features and auxiliary generator benefit warm-up training in FETA-Pro? **(RQ3)** How do the hyper-parameters of FETA-Pro affect the synthesis performance?

5.1 Performance of Synthetic Datasets (RQ1)

This RQ investigates the utility and fidelity of synthetic images from FETA-Pro and baselines. We compare FETA-Pro with seven baselines on five investigated image datasets as described in Section 4, under the privacy budget $\varepsilon = \{1, 10\}, \delta = 1/(N_{\text{priv}} \times \log N_{\text{priv}})$. Figure 3 visualizes synthetic image samples generated by FETA-Pro and baselines under $\varepsilon = 10$, alongside real images. More details of privacy budget allocation plans are shown in Appendix C.4 [39].

Table 4 presents the FID and Acc (%) of FETA-Pro and baselines. In Table 4, we observe that FETA-Pro significantly outperforms the performance of baselines in terms of both utility and fidelity. Specifically, for the CIFAR-10 and Camelyon datasets under $\varepsilon = 1$, FETA-Pro achieves FID scores of 120.4 and 31.0, respectively, markedly surpassing the 139.8 and 52.8 scores of DP-FETA. Under $\varepsilon = 10$, FETA-Pro attains an FID score of 69.0 on the CIFAR-10 dataset, outperforming the FID of 95.3 achieved by DP-FETA. Then, the accuracy improves from 43.3% to 47.0%. We present more comparisons of fidelity metrics, including Inception Score [46], Precision and Recall [47], and Fréchet Leakage Distance [48], for FETA-Pro and baseline methods in Appendix D.1 [39].

We track the FID of synthetic images during DP-SGD fine-tuning to assess convergence against baseline methods. Referring to the previous implementation in DPImageBench [4], we sample 6,000 images (10% of the final synthetic dataset) to compute the intermediate results of FID relative to the sensitive dataset. Figure 4 depicts the FID of synthetic images across various fine-tuning iterations on sensitive images, benchmarked against baseline methods. The figure reveals that synthetic images of FETA-Pro converge more rapidly than those of other baselines, enhancing the fidelity of synthetic images. Additionally, compared to DP-GAN, which uses a GAN-based synthesizer, using a diffusion model as the synthesizer offers greater stability.

Answers to RQ1: Synthetic images produced by FETA-Pro show superior fidelity and utility compared to baselines under $\varepsilon = 1$. On average, the FID of synthetic images generated by FETA-Pro is 25.7% lower, and the Acc on downstream classification tasks is 4.1% higher than the SOTA method DP-FETA. The FID of synthetic images of FETA-Pro converges more rapidly than that of baselines during training.

5.2 Benefits of Frequency Features and Auxiliary Generator (RQ2)

We explore the strengths of leveraging the frequency features to warm up the DP image synthesizers. We compare the performance of FETA-Pro with three invariants of FETA-Pro under $\varepsilon = 1.0$.

- 'FETA-Pro$_{ft}$' denotes a synthesizer that prioritizes learn-

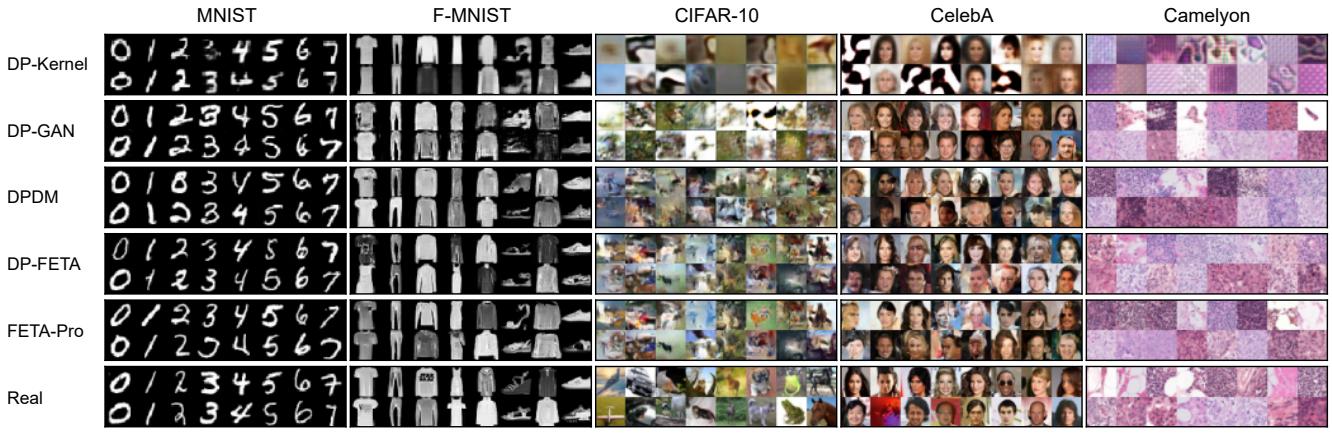


Figure 3: Synthetic Image Examples under $\varepsilon = 10$. The last row of images is real image samples from sensitive image datasets. Due to space constraints, we showcase only the top-4 performing baseline methods: DP-Kernel, DP-GAN, DPDM, and DP-FETA.

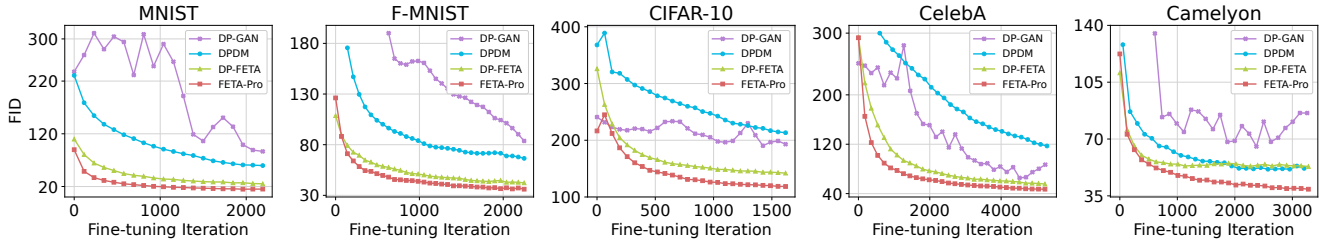


Figure 4: FID of synthetic images during fine-tuning, compared to baseline methods to evaluate convergence under $\varepsilon = 1$. The FID is computed using 6,000 sampled images (10% of the final synthetic dataset) relative to the sensitive dataset.

ing **frequency** knowledge from sensitive datasets before acquiring **spatial** knowledge.

- 'FETA-Pro$_{mix}$' means the synthesizers simultaneously learn the spatial and frequency knowledge.
- 'FETA-Pro$_f$' means the synthesizers solely learn the frequency knowledge for warm-up.

To thoroughly explore the benefits of warm-up, this RQ examines three perspectives: (1) Is warm-up necessary? (2) Which features are most effective for warm-up? (3) How should the selected features be used for warm-up?

Figure 5 presents the FID and Acc of FETA-Pro and its variants. For the first question, this figure shows that the training shortcuts enhance synthetic performance. FETA-Pro achieves FID reductions of 77.8% $(= (36.1 - 8.0/36.1) \times 100\%)$, 47.9% $(= (53.5 - 27.8/53.5) \times 100\%)$, 68.8% $(= (153.9 - 48.0/153.9) \times 100\%)$, and 72.3% $(= (111.9 - 31.0/111.9) \times 100\%)$ compared to the no warm up method DPDM.

For the second question, this figure highlights that frequency features outperform spatial features for synthesizer warm-up, as analyzed in Section 2.3. Specifically, FETA-Pro$_f$ achieves FID of 9.8, 31.1, 58.0, and 45.2, while DP-FETA, using only spatial features for warm-up, attains reductions of 13.7, 31.4, 60.2, and 52.8 across four datasets. Additionally, FETA-Pro$_f$ outperforms DP-FETA in accuracy, producing higher-quality synthetic images. By combining both feature types, FETA-Pro achieves the best synthetic performance among the compared methods.

For the third question, FETA-Pro$_{mix}$, which learns frequency and spatial features simultaneously, exhibits poorer synthetic performance than FETA-Pro and FETA-Pro$_{ft}$, which learn these features separately, in most cases. Figure 5 further presents that synthetic images of FETA-Pro$_{ft}$ generally outperform DP-FETA in both utility and fidelity, though they fall short of FETA-Pro's performance. For instance, on the `Camelyon` dataset, FETA-Pro$_{ft}$ achieves an FID of 37.7 and an accuracy of 83.3%, outperforming DP-FETA's FID of 52.8 and accuracy of 77.3%, under $\varepsilon = 1$. These results verify the analysis introduced in Section 3.3. Integrating both spatial and frequency features enhances DP synthesis. Additionally, frequency features are more intricate than their spatial counterparts, and benefit from the 'from easy to hard' learning paradigm. Prioritizing learning simpler features (spatial features) before learning complex features (frequency features) benefits DP synthesis.

To investigate the benefit of auxiliary generators, Table 5 compares FETA-Pro with two methods mentioned in Sec-

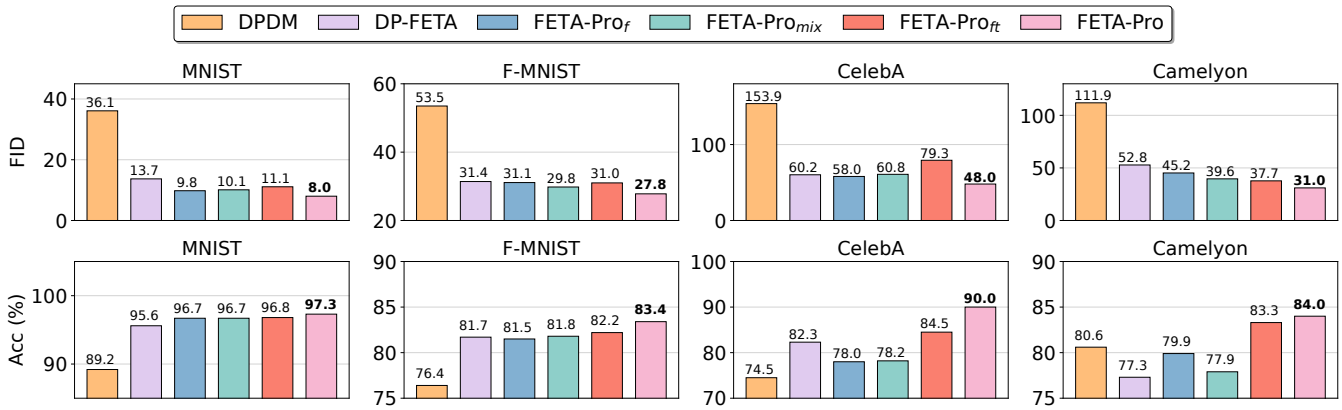


Figure 5: FID (top row) and Acc (bottom row) of FETA-Pro and five baselines with $\varepsilon = 1$. 'DPDM' indicates no warm-up. 'DP-FETA' and 'FETA-Pro$_f$' use only spatial and frequency features for warm-up, respectively. 'FETA-Pro$_{mix}$' learns spatial and frequency features simultaneously. 'FETA-Pro$_{ft}$' first learns frequency domain features, then spatial features. 'FETA-Pro' is our work, which learns spatial domain features, then frequency features.

Table 5: Performance of FETA-Pro on five sensitive datasets with $\varepsilon = 1$. 'FETA-Pro-No-Auxiliary' means directly warming up synthesizers on frequency features. 'FETA-Pro-DM-Auxiliary' means using diffusion models as the auxiliary generator. 'GPU memory' means the peak GPU memory usage across all stages.

Dataset	FETA-Pro				FETA-Pro-No-Auxiliary				FETA-Pro-DM-Auxiliary			
	FID	Acc (%)	Time	GPU Memory	FID	Acc (%)	Time	GPU Memory	FID	Acc (%)	Time	GPU Memory
MNIST	8.0	97.3	11.7 H	73.7 GB	36.5	90.6	13.4 H	74.8 GB	23.1	92.7	13.5 H	73.7 GB
F-MNIST	27.8	83.4	11.7 H	73.7 GB	60.5	77.2	13.4 H	74.8 GB	47.3	78.4	13.5 H	73.7 GB
CIFAR-10	120.4	37.9	20.7 H	96.3 GB	237.6	29.8	27.8 H	135.8 GB	230.7	29.9	27.9 H	96.3 GB
CelebA	48.0	90.0	54.4 H	96.3 GB	260.1	67.0	68.3 H	135.8 GB	92.7	78.4	68.4 H	96.3 GB
Camelyon	31.0	84.0	32.9 H	96.3 GB	62.4	80.5	47.6 H	135.8 GB	51.8	82.4	47.8 H	96.3 GB

tion 3.1. 'FETA-Pro-No-Auxiliary' means directly warming up synthesizers on frequency features without leveraging auxiliary generators, as introduced in Appendix C.1. 'FETA-Pro-DM-Auxiliary' means using diffusion models as the auxiliary generator. Table 5 shows that FETA-Pro-No-Auxiliary exhibits the worst synthetic performance, while auxiliary generators significantly enhance performance. In addition, FETA-Pro outperforms FETA-Pro-DM-Auxiliary in terms of both performance and efficiency, showing that GANs are more effective than diffusion models as auxiliary generators.

Answers to RQ2: Using frequency features for synthesizer warm-up proves more effective than using spatial features. Besides, combining both feature types enhances feature complementarity, achieving improvements of 3.6% in Acc and 10.1 in FID compared to using only frequency features across the studied datasets. Besides, we should prioritize learning spatial features before learning frequency features for better synthetic performance.

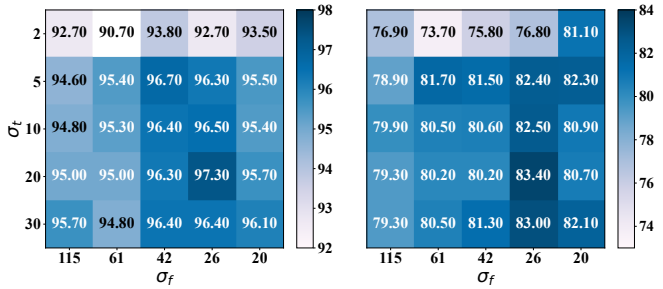
5.3 The Impact of Hyper-Parameters (RQ3)

This RQ investigates the impact of hyper-parameter settings from two perspectives as follows. We conduct experiments both under the privacy budget $\varepsilon = 1$ in this section.

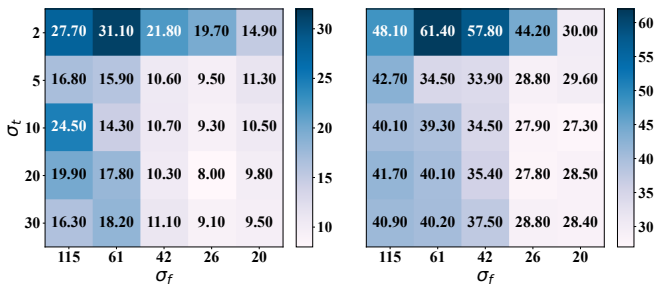
- **Privacy allocations.** We evaluate FETA-Pro under different privacy allocation plans. As introduced in Section 3.4, FETA-Pro consumes privacy budget across three stages: (1) extracting spatial features and (2) frequency features from sensitive datasets; and (3) fine-tuning synthesizers on sensitive datasets leveraging DP-SGD. We consider the combination of the noise scales (e.g., σ_t and σ_f) $\{2, 5, 10, 20, 30\}$ and $\{20, 26, 42, 61, 115\}$ for spatial and frequency features. A low level of noise scale means a high level of privacy budget allocation. We ensure that the total privacy budget for these three processes is constrained to $\varepsilon = 1$. Appendix D.2 [39] presents the noise scale of DP-SGD under different privacy allocation strategies.

- **Privacy budget.** We assess the utility and fidelity of synthetic images across six privacy budgets, $\varepsilon \in \{0.2, 1.0, 5.0, 10.0, 15.0, 20.0\}$. We adopt the same privacy allocation ratio as described in Appendix D.2 [39].

Figure 6 presents the synthetic performance under different DP privacy budget allocation strategies. We observe that, given the same spatial domain budget allocation, a higher frequency domain budget (low noise scale) allocation generally



(a) Acc of synthetic images for MNIST (left) and F-MNIST (right).



(b) FID of synthetic images MNIST (left) and F-MNIST (right).

Figure 6: The Acc and FID of synthetic MNIST and F-MNIST images under $\varepsilon = 1$ and $\delta = 1/N\log(N)$. We explore combinations of noise scales $\sigma_t \in \{2,5,10,20,30\}$ and $\sigma_f \in \{20,26,42,61,115\}$ (privacy budget from high to low) for spatial and frequency features, ensuring the total privacy budget across all processes (include DP-SGD) is constrained to $\varepsilon = 1$.

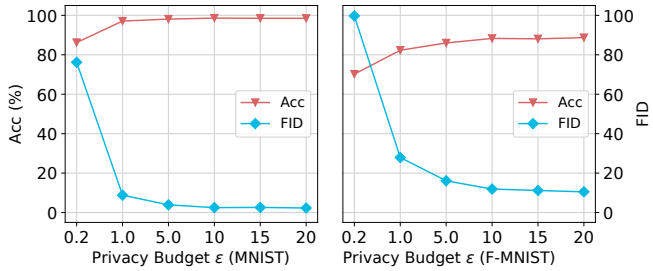


Figure 7: The Acc and FID of synthetic images generated by FETA-Pro for MNIST and F-MNIST under privacy budgets $\varepsilon = \{0.2, 1.0, 5.0, 10, 15, 20\}$ and a fixed $\delta = 1/N\log(N)$.

achieves superior synthetic performance. Specifically, in the F-MNIST dataset with a spatial domain noise level of $\sigma_t = 2$, the Acc decreases from 81.1 to 76.9 as the frequency domain noise level σ_f increases from 20 to 115. Overall, the synthetic performance is sensitive to the value of privacy allocation strategies. In terms of Acc, the best and worst results in Figure 6 of differences are 6.6% $(= (97.3 - 90.7) \times 100\%)$ and 9.7% $(= (83.4 - 73.7) \times 100\%)$ for MNIST and F-MNIST. In terms of FID, the differences are 23.1 $(= (31.1 - 8.0))$ and 34.1 $(= (61.4 - 27.3))$. Generally, overly low values of σ_t lead to reduced synthetic performance. For instance, in the MNIST dataset, when $(\sigma_t = 2, \sigma_f = 20)$, FETA-Pro achieves an accuracy of 93.5%, which is noticeably lower than the 97.3% obtained with $(\sigma_t = 20, \sigma_f = 26)$. As presented in Table 12 of Appendix D.2 [39], overly low values of σ_t indicate that larger privacy budgets are allocated to spatial and domain features, while a smaller budget is assigned to DP-SGD. Generally, privacy budget ratios ordered as 'spatial features < frequency features < DP-SGD' can obtain better synthetic performance (referring to the privacy budget ratios in Table 12 of Appendix D.2 [39]). We provide the relatively optimal combinations of σ_f and σ_t values in Table 9 of Appendix C.4 [39] for different sensitive datasets.

In terms of Acc, FETA-Pro achieves optimal results of 97.3% and 83.4% for the MNIST and F-MNIST datasets, respectively, at $(\sigma_t = 20, \sigma_f = 26)$. In terms of FID, FETA-Pro achieves optimal results of 8.0 and 27.3, at $(\sigma_t = 20, \sigma_f = 26)$ with a spatial-to-frequency budget cost ratio of $0.3/2.92$ and $(\sigma_t = 10, \sigma_f = 20)$ with a ratio of $1.22/4.99$, as presented in Table 12 of Appendix D.2 [39]. These results indicate that superior performance is achieved by allocating a greater privacy budget to frequency features compared to spatial features.

Figure 7 presents the Acc and FID scores under different privacy budgets. We observe that a higher privacy budget achieves improved synthetic performance. For the MNIST dataset, when $\varepsilon \geq 10$, the Acc approaches 100%, and the FID nears 0, indicating that the synthetic images closely resemble the original ones. Similarly, for F-MNIST, performance plateaus at $\varepsilon \geq 10$, suggesting that $\varepsilon = 10$ balances synthetic performance with privacy preservation in practice.

Answers to RQ3: Allocation strategies highly affect synthetic image quality. Overall, privacy budget ratios in FETA-Pro ordered as 'spatial features < frequency features < DP-SGD' can obtain better performance. For both MNIST and F-MNIST, performance stabilizes at $\varepsilon \geq 10$, indicating that $\varepsilon = 10$ balances synthetic quality with privacy preservation in practice.

6 Discussions

This section discusses (1) the impact of the DP on the performance of FETA-Pro, (2) how FETA-Pro performs when using public images, and (3) the computational resources requirements. Appendix E [39] introduces the limitations.

6.1 FETA-Pro without Privacy Protection

This experiment evaluates the impact of the DP on the synthetic performance of FETA-Pro. We compare FETA-Pro with three baseline methods: (1) '$\varepsilon = \infty$' means that diffusion models are trained using the FETA-Pro framework without

Table 6: FID and Acc (%) of FETA-Pro on four sensitive datasets with $\varepsilon = \{10, \infty\}$. 'No DP' means the classifier is trained directly on the sensitive dataset.

Method	MNIST		F-MNIST		CelebA		Camelyon	
	FID	Acc	FID	Acc	FID	Acc	FID	Acc
No DP	-	99.7	-	94.5	-	97.7	-	87.7
Ours ($\varepsilon = 10$)	2.5	98.6	11.6	88.1	20.0	95.2	20.3	84.3
Ours ($\varepsilon = \infty$)	1.1	99.3	5.7	91.0	7.0	95.3	6.6	86.3

introducing Gaussian noise during training. (2) 'No DP': The classifier is trained directly on the sensitive dataset for the downstream classification task. Note that the FID evaluates the quality of images generated by generative models; thus, sensitive datasets do not have an FID score.

Table 6 shows that, under a privacy budget of $\varepsilon = 10$, FETA-Pro achieves average accuracy (Acc) reductions of only 0.7% $(= (99.3 - 98.6) \times 100\%)$, 2.9% $(= (91.0 - 88.1) \times 100\%)$, 0.1% $(= (95.3 - 95.2) \times 100\%)$, and 2.0% $(= (86.3 - 84.3) \times 100\%)$ across four sensitive datasets compared to the non-private setting ($\varepsilon = \infty$). As highlighted in previous work [49], it is common to use $\varepsilon \leq 10$ in differential practical synthesis tasks. Therefore, FETA-Pro generates highly useful synthetic images while protecting sensitive data. Besides, the FID decreases by 1.4 $(= (2.5 - 1.1))$, 5.9 $(= (11.6 - 5.7))$, 13.0 $(= (20.0 - 7.0))$, and 13.7 $(= (20.3 - 6.6))$. Compared to directly using sensitive datasets for downstream tasks (i.e., 'No DP' in Table 6), FETA-Pro exhibits an average Acc reduction of 3.4%. These results indicate the need for further improvements to FETA-Pro.

6.2 FETA-Pro Leveraging Public Images

This section evaluates whether pre-training with public datasets enhances the synthetic performance of FETA-Pro and compares FETA-Pro with state-of-the-art methods using public images, as detailed below. Referring to DPImageBench [4], we leverage `ImageNet` [50] as the pre-training dataset.

- **PDP-Diffusion [28]:** PDP-Diffusion adopts the 'public pre-training + private fine-tuning' paradigm. It incorporates large batch sizes to improve the stability and accelerate the convergence of diffusion model training under DP-SGD. Moreover, leveraging public datasets for pre-training allows the synthesizer to benefit from a broader knowledge base.
- **PrivImage [5]:** Unlike PDP-Diffusion, which uses entire public datasets for pre-training, PrivImage selectively queries the semantic distribution of sensitive data to identify a subset of public data, narrowing the distribution gap between public and sensitive datasets and enhancing the efficiency of pre-training [5].
- **Private Evolution (PE) [6]:** PE progressively guides the pretrained models (either local or blackbox models behind APIs) to generate synthetic images resembling a sensitive dataset, eliminating the need for training or fine-tuning.

Table 7: FID and Acc of FETA-Pro and baselines which use public images on four image datasets with $\varepsilon = 1$. The best performance in each column is highlighted using a bold font.

Method ($\varepsilon = 1$)	MNIST		F-MNIST		CelebA		Camelyon	
	FID	Acc	FID	Acc	FID	Acc	FID	Acc
PDP-Diffusion	8.9	94.5	16.6	79.2	17.2	89.4	**15.0**	**85.2**
PrivImage	**7.6**	94.0	**16.1**	79.9	**12.3**	**90.8**	15.2	82.8
PE	48.0	27.9	48.8	47.9	23.0	70.5	71.5	63.3
FETA-Pro	8.0	**97.3**	27.8	**83.4**	48.0	90.0	31.0	84.0
PDP-Diffusion-Pro	10.9	95.3	23.4	81.3	25.2	90.1	17.2	84.1
PrivImage-Pro	9.9	94.7	25.4	78.2	23.1	90.2	15.9	84.4

Table 7 compares the FID and accuracy of FETA-Pro and baseline methods using public images across four image datasets with $\varepsilon = 1$. The 'PDP-Diffusion-Pro' and 'PrivImage-Pro' variants incorporate the warm-up method from FETA-Pro into PDP-Diffusion [28] and PrivImage [5], respectively. These two variant methods follow the 'public pre-training + private warm-up + private fine-tuning' paradigm. FETA-Pro is our proposed method in this work.

Comparing the results in rows four through six of Table 7, we observe that public pre-training enhances synthetic performance on `CelebA` and `Camelyon` datasets, but reduces performance on `MNIST` and accuracy on `F-MNIST`. Overall, warm-ups do not always benefit the algorithms. Specifically for the `MNIST` dataset, FETA-Pro achieves an FID of 8.0 and an accuracy of 97.3%. This performance surpasses methods that utilize public resources, such as PDP-Diffusion (FID: 8.9, Acc: 94.5%) and PE (FID: 48.0, Acc: 27.9%). Although the FID achieved by FETA-Pro is 0.4 lower than PrivImage (8.0 vs. 7.6), its Acc is 3.3 percentage points higher (97.3% vs. 94.0%). Pre-training on public datasets may disrupt FETA-Pro's warm-up process, leading to degraded performance. Specifically, the FID increases from 8.0 to 10.9, and the accuracy drops from 97.3% to 95.3% compared to PDP-Diffusion-Pro.

For `CelebA` and `Camelyon`, pre-training enhances synthetic performance so that PDP-Diffusion-Pro and PrivImage-Pro result in lower FID and higher accuracy compared to FETA-Pro. While PDP-Diffusion-Pro and PrivImage-Pro show improved synthetic performance compared to FETA-Pro, their comparison against PDP-Diffusion and PrivImage achieves mixed results. We can observe that in `CelebA`, PrivImage-Pro achieves better Acc (90.2%) than PDP-Diffusion (89.4%). However, PDP-Diffusion-Pro and PrivImage-Pro achieve higher (and thus worse) FID scores than both PDP-Diffusion and PrivImage. Their best accuracies (PrivImage-Pro: 90.2% for `CelebA`, 84.4% for `Camelyon`) still do not surpass the peak performance observed in PDP-Diffusion (85.2% for `Camelyon`) and PrivImage (90.8% for `CelebA`). Overall, we advise against combining pre-training with the warm-up, as these operations may interact detrimentally, leading to suboptimal results.

Table 8: GPU memory usage and runtime analysis of studied methods for the CIFAR-10. 'GPU memory' means the peak GPU memory usage across all stages.

Algorithm	Stage	Memory	Runtime	GPU Memory
DP-MERF	Warm-up	0 GB	0 H	18.6 GB
	Fine-tune	3.5 GB	0.02 H	
	Synthesis	18.6 GB	0.03 H	
DP-NTK	Warm-up	0 GB	0 H	21.3 GB
	Fine-tune	4.8 GB	7.75 H	
	Synthesis	19.1 GB	0.05 H	
DP-Kernel	Warm-up	0 GB	0 H	22.5 GB
	Fine-tune	8.8 GB	3.6 H	
	Synthesis	19.0 GB	0.05 H	
GS-WGAN	Warm-up	21.6 GB	18.3 H	21.6 GB
	Fine-tune	21.6 GB	0.6 H	
	Synthesis	20.0 GB	0.01 H	
DP-GAN	Warm-up	0 GB	0 H	46.1 GB
	Fine-tune	46.1 GB	3.25 H	
	Synthesis	20.1 GB	0.03 H	
DPDM	Warm-up	0 GB	0 H	96.3 GB
	Fine-tune	96.3 GB	20.5 H	
	Synthesis	32.1 GB	0.17 H	
DP-FETA	Warm-up	3.4 GB	0.04 H	96.3 GB
	Fine-tune	96.3 GB	20.5 H	
	Synthesis	32.1 GB	0.17 H	
FETA-Pro	Warm-up	3.4 GB	0.1 H	96.3 GB
	Fine-tune	96.3 GB	20.5 H	
	Synthesis	32.1 GB	0.17 H	

6.3 Comparison of Computational Resources

This section investigates the computational resource usage of various methods. The parameter sizes of DP image synthesizers and the servers used for implementation are detailed in Section 4. Table 8 presents the GPU memory and runtime usage of baselines and FETA-Pro. Compared to the SOTA DP image synthetic method DP-FETA, which only uses spatial features to warm-up, FETA-Pro only introduces an average of 0.3% additional training-time cost (i.e., 0.06 H) for frequency features warming up while bringing increases in fidelity and utility metrics as presented in Table 4. Additionally, FETA-Pro incurs no extra memory overhead compared to DP-FETA.

7 Related Work

We discuss related work briefly here, and we provide a more comprehensive discussion in Appendix F [39]. We discuss previous DP image synthesis methods based on whether the methods leverage public resources or not.

Without using Public Resources. These methods train the synthesizer solely using the sensitive image dataset, without relying on external public resources [15, 19, 24, 51]. A group of methods proposes adding noise to the high-level feature for DP, e.g., random Fourier features [51, 52], empirical neural tangent kernels [24], of sensitive images, and training a synthesizer to ensure synthetic images approach noisy features.

Although the aforementioned methods have low computational requirements, they present suboptimal synthetic performance on complex datasets, such as CelebA [22]. Abadi et al. [14] introduce DP-SGD, which adds noise to training gradients to achieve DP. Various works use DP-SGD to train deep generative models on sensitive datasets for addressing DP synthesis of complex images [5, 9, 10, 26–28, 41]. Among these approaches, utilizing diffusion models as synthesizers achieves SOTA synthetic performance [5, 9, 27, 28].

Using Public Dataset. Several methods suggest using a publicly available dataset to initially train a data synthesizer, followed by fine-tuning models with sensitive images using DP-SGD, to improve synthetic performance [9, 10, 28, 52–55]. For example, Ghalebikesabi et al. [28] apply this pre-training and fine-tuning method using diffusion models. Li et al. [5] propose PrivImage, which enhances synthetic data utility by selecting a subset of public data that aligns distributionally with the sensitive dataset. This paradigm represents the predominant framework for DP image synthesis.

Using Pretrained Models. Fine-tuning with DP-SGD remains computationally intensive and is time-consuming [4]. Lin et al. [6] propose a fine-tuning-free method, Private Evolution (PE). This approach iteratively directs pre-trained models to produce synthetic images that closely align with sensitive images in the feature space. However, when the distribution of synthetic data generated by the pre-trained models diverges from that of the sensitive data, PE's performance underperforms compared to fine-tuning-based methods [4, 7]. To address this limitation, Lin et al. [7] introduce Sim-PE, an extension of PE that uses powerful non-neural-network simulators (e.g., avatar generator [56], computer graphics tools) instead of pre-trained models to fit sensitive data under DP.

8 Conclusions

This paper investigates the incorporation of different training shortcuts to augment DP-SGD. FETA-Pro integrates frequency features, which robustly capture global structures and textures, as the training shortcut from two key perspectives: (1) they complement spatial features, and (2) their complexity, falling between that of spatial features and full images, enables a fine-grained curriculum framework. This allows FETA-Pro to learn sequentially from 'spatial features $\rightarrow$ frequency features $\rightarrow$ images.' When combining these two types of shortcuts, we should address the challenges of training discrepancy between spatial and frequency features. To do so, instead of training one model on multiple features or objectives, we leverage the pipeline generation property of generative models. This allows us to use a more flexible design by having multiple models work on different features. Specifically, FETA-Pro introduces an *auxiliary generator* to produce images from noisy frequency features. Then, another model is trained with these generated images, alongside spatial features and DP-SGD. Thus, FETA-Pro represents both feature types

in a unified manner. We conduct experiments across five sensitive image datasets, showing that FETA-Pro achieves better synthetic images compared to the state-of-the-art methods. By integrating the strengths of multiple synthesizers, FETA-Pro surpasses traditional single-synthesizer approaches, offering new insights for future research in DP image synthesis.

Acknowledgment

We thank anonymous reviewers for their valuable comments. This paper was partially supported by NSF CNS-2213700, NSF CCF-2220433. The opinions, findings, conclusions, and recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Ethical Considerations

We structure the ethical considerations discussion by linking our *stakeholder analysis* to the impacts generated during two distinct phases: the *research process* (data handling) and the *publication of results* (deployment and application). We then detail *mitigations* (specifying which stakeholder group each measure protects) and conclude with the *justification* for conducting this research.

Stakeholder Analysis and Process Impact. FETA-Pro involves three primary stakeholder groups as follows. (1) *Data Subjects*: These are the individuals represented in sensitive datasets (e.g., medical scans, faces). During the research process, these subjects rely on requirements to confidentiality protocols to ensure their sensitive visual data is not exposed. (2) *Data Owners (Institutions)*: Entities such as hospitals and research institutions hold these sensitive datasets. Throughout the research process, these entities face challenges due to privacy regulations and legal restrictions that hinder collaboration. (3) *Researchers and Practitioners:* This group includes the ML and security community. They rely on the process to produce methodologically sound and reproducible DP.

Impact of the Research. The publication of FETA-Pro has both positive and negative impacts, affecting the stakeholders.

Positive Impacts. (1) Facilitating Data Sharing (Impact on Data Owners & Researchers): FETA-Pro improves the fidelity of DP synthetic images. We provide data owners with a tool to share data utility without violating privacy regulations. This directly benefits Researchers by enabling access to previously inaccessible data for training ML models. (2) Reproducibility and Open Science (Impact on Practitioners): We have open-sourced our code and provided detailed explanations to ensure reproducibility. This allows the practitioner community to responsibly assess, audit, and improve upon our methods, ensuring that future applications in real-world scenarios are built on a transparent foundation.

Negative Impacts. (1) Bias Propagation (Impact on Data Subjects): We acknowledge that DP synthetic images may inherit or even amplify biases present in the original data (e.g., gender or racial biases). If practitioners train models on synthetic images, the resulting systems may exhibit reduced fairness. This negatively impacts data subjects, who may be affected by decisions made by models with ingrained biases. (2) Potential for Misuse (Impact on Data Subjects & Society): The publication of high-fidelity synthesis methods carries the risk of misuse. Malicious actors could use these methods to generate realistic fake content (forging identities or fraud) or to train invasive surveillance systems.

Mitigation. To address the risks outlined above, we have implemented mitigation strategies. We clarify below how each technical measure serves to protect a specific stakeholder.

Methodological Mitigations (Implemented). (1) Protecting Data Subjects via Data Curation: To safeguard data subjects from the amplification of external biases, we avoid using public datasets for pretraining [11]. Public datasets may contain uncontrolled biases (e.g., category imbalances) or inadvertent sensitive content. By restricting pretraining, we minimize the risk that our synthetic images inherit prejudices. (2) Ensuring Reliability for Practitioners via Validation: We incorporate rigorous validation steps to detect potential inaccuracies in synthetic images. While this ensures technical correctness, its primary ethical function is to protect practitioners from drawing false conclusions. By preventing the downstream propagation of flawed data, we ensure that models built on our synthesis do not result in erroneous decisions. (3) Empowering the Community via Transparency: We release our code. This empowers the research community to audit our privacy claims and allows institutions to verify the safety before applying it to their own sensitive data.

Recommended Future Deployment Measures. Beyond our methodological choices, we advocate for the following measures to protect stakeholders in reality. (1) Fairness Audits for Data Subjects: To further protect data subjects from algorithmic discrimination, we recommend that practitioners integrate fairness constraints [57]. The evaluation should be expanded to include comprehensive fairness metrics [58] to ensure that the synthetic data does not marginalize specific demographic groups. (2) Safeguards Against Misuse for Society: To address the societal risk of deepfakes, we suggest the implementation of DP watermarking [59]. Inserting indelible watermarks into synthetic datasets allows tracing the origin of the data, serving as a deterrent against malicious actors.

Justification for Research. This work addresses a critical challenge in privacy-preserving machine learning: balancing utility and privacy without relying on risky public datasets. Conducting this research is essential for the following reasons: (1) FETA-Pro presents a novel integration of spatial and frequency-domain features, contributing new insights into the design for privacy-preserving generative models; (2) We

share the code of our method to promote transparency and community-driven improvements, reducing the risk of opaque implementations. Our repository also serves as a step towards identifying potential risks in DP image synthesis.

Open Science

We release the replication package on the anonymous link,[3]. In the `README.md` files of the repository, we provide a clear, point-by-point explanation that maps each table and figure to the corresponding code needed to generate its results. The DOI for the artifacts is on Zenodo.[4]

We provide a detailed description of how our repository matches each table and figure in our paper as follows.

Environment: We provide a straightforward environment setup. The user only needs to run "bash install.sh" to install our environment in minutes, supporting availability and functionality assessments.

Data Preparation: This artifact provides a concrete data preparation process. All datasets used are publicly available and widely used in the community to avoid ethical concerns. The user should run "bash data_preparation.sh" to prepare all the studied datasets. Loading all datasets is time-consuming. For quick evaluation, we recommend loading only `MNIST` and `Fashion-MNIST`, using "bash data_preparation_quick.sh".

Main Contributions: Our artifices provides the reproducibility of tables and figures presented in the main body as follows.

- `Table 4`: In RQ1, FID and Acc (%) of FETA-Pro and seven baselines on `MNIST`, `F-MNIST`, `CIFAR-10`, `CelebA` and `Camelyon` with $\varepsilon = \{1, 10\}$.
- `Table 5`: In RQ2, performance of FETA-Pro on five sensitive datasets with $\varepsilon = 1$, compared to 'FETA-Pro-No-Auxiliary' and 'FETA-Pro-DM-Auxiliary' (the variants of FETA-Pro).
- `Table 6`: In Discussion, FID and Acc (%) of FETA-Pro on four sensitive datasets with $\varepsilon = \{10, \infty\}$. 'No DP' means the classifier is trained directly on the sensitive dataset.
- `Table 7`: In Discussion, FID and Acc of FETA-Pro and baselines (PDP-Diffusion and PrivImage) which use public images on four image datasets with $\varepsilon = 1$.
- `Table 10`: In Appendix D.1 [39], the IS, Precision, Recall, and FLD of FETA-Pro and seven baselines on five studied datasets with $\varepsilon = \{1, 10\}$.
- `Table 13`: In Appendix D.2 [39], FID and Acc (%) of FETA-Pro on five sensitive datasets with $\varepsilon = 10$ using RDP and PRV privacy budget accounting methods.
- `Table 12`: In Appendix D.2 [39], RDP cost ratios (%) of spatial features / frequency features / DP-SGD in FETA-Pro using various privacy allocation strategies ($\varepsilon = 1.0$).
- `Figure 3`: In RQ1, synthetic image examples under $\varepsilon = 10$.
- `Figure 4`: In RQ1, FID of synthetic images during fine-tuning, compared to baseline methods to evaluate convergence under $\varepsilon = 1$.
- `Figure 5`: In RQ2, FID and Acc of FETA-Pro and five baselines with $epsilon = 1$. 'DPDM' indicates no warm-up. 'DP-FETA' and 'FETA-Pro$_f$' use only spatial and frequency features for warm-up, respectively. 'FETA-Pro$_{mix}$' learns spatial and frequency features simultaneously. 'FETA-Pro$_{ft}$' first learns frequency domain features, then spatial features. 'FETA-Pro' is our work.
- `Figure 6`: In RQ3, Acc and FID of synthetic `MNIST` and `F-MNIST` images under $\varepsilon = 1$ and different privacy budget allocation plans.
- `Figure 7`: In RQ3, The Acc and FID of synthetic images generated by FETA-Pro for `MNIST` and `F-MNIST` under privacy budgets $\varepsilon = \{0.2, 1.0, 5.0, 10, 15, 20\}$.

[3] https://github.com/2019ChenGong/Feta-Pro
[4] https://doi.org/10.5281/zenodo.17836341

References

[1] Y. Hu, F. Wu, Q. Li, *et al.*, "Sok: Privacy-preserving data synthesis," in *IEEE Symposium on Security and Privacy (SP)*, pp. 2–2, 2024.

[2] Z. Zhang, T. Wang, N. Li, *et al.*, "{PrivSyn}: Differentially private data synthesis," in *USENIX Security Symposium*, pp. 929–946, 2021.

[3] D. Sun, J. Q. Chen, C. Gong, T. Wang, and Z. Li, "Netdpsyn: synthesizing network traces under differential privacy," in *Proceedings of the 2024 ACM on Internet Measurement Conference*, pp. 545–554, 2024.

[4] C. Gong, K. Li, Z. Lin, and T. Wang, "Dpimagebench: A unified benchmark for differentially private image synthesis," *arXiv preprint arXiv:2503.14681*, 2025.

[5] K. Li, C. Gong, Z. Li, *et al.*, "PrivImage: Differentially private synthetic image generation using diffusion models with Semantic-Aware pretraining," in *33rd USENIX Security Symposium*, pp. 4837–4854, 2024.

[6] Z. Lin, S. Gopi, J. Kulkarni, *et al.*, "Differentially private synthetic data via foundation model APIs 1: Images," in *The Twelfth International Conference on Learning Representations*, 2024.

[7] Z. Lin, T. Baltrusaitis, and S. Yekhanin, "Differentially private synthetic data via apis 3: Using simulators instead of foundation model," *arXiv preprint arXiv:2502.05505*, 2025.

[8] H. Wang, Z. Lin, D. Yu, and H. Zhang, "Synthesize privacy-preserving high-resolution images via private textual intermediaries," *arXiv preprint arXiv:2506.07555*, 2025.

[9] M. F. Liu, S. Lyu, M. Vinaroz, and M. Park, "Differentially private latent diffusion models," *arXiv preprint arXiv:2305.15759*, 2023.

[10] Y.-L. Tsai, Y. Li, Z. Chen, *et al.*, "Differentially private fine-tuning of diffusion models," *arXiv preprint arXiv:2406.01355*, 2024.

[11] F. Tramèr, G. Kamath, and N. Carlini, "Position: Considerations for differentially private learning with large-scale public pretraining," in *Forty-first International Conference on Machine Learning*, 2024.

[12] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, *et al.*, "Extracting training data from large language models," in *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

[13] B. Wang, W. Chen, H. Pei, C. Xie, M. Kang, C. Zhang, C. Xu, Z. Xiong, R. Dutta, R. Schaeffer, *et al.*, "Decodingtrust: A comprehensive assessment of trustworthiness in gpt models.," in *NeurIPS*, 2023.

[14] M. Abadi, A. Chu, I. J. Goodfellow, and et al., "Deep learning with differential privacy," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 308–318, 2016.

[15] K. Li, C. Gong, X. Li, Y. Zhao, X. Hou, and T. Wang, "From easy to hard: Building a shortcut for differentially private image synthesis," in *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 3656–3674, IEEE Computer Society, 2025.

[16] J. Crabbé, N. Huynh, J. Stanczuk, and M. Van Der Schaar, "Time series diffusion in the frequency domain," in *Proceedings of the 41st International Conference on Machine Learning*, pp. 9407–9438, 2024.

[17] K. Li, Z. Huang, X. Hou, and C. Hong, "Gaussmarker: Robust dual-domain watermark for diffusion models," *arXiv preprint arXiv:2506.11444*, 2025.

[18] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 9, pp. 4555–4576, 2021.

[19] F. Harder, K. Adamczewski, and M. Park, "DP-MERF: differentially private mean embeddings with random features for practical privacy-preserving data generation," in *AISTATS*, pp. 1819–1827, 2021.

[20] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, 2020.

[21] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, and et al., "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014.

[22] Z. Liu, P. Luo, X. Wang, and et al., "Deep learning face attributes in the wild," in *2015 IEEE International Conference on Computer Vision*, pp. 3730–3738, 2015.

[23] M. Heusel, H. Ramsauer, T. Unterthiner, *et al.*, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[24] Y. Yang, K. Adamczewski, and et al, "Differentially private neural tangent kernels for privacy-preserving data generation," *CoRR*, vol. abs/2303.01687, 2023.

[25] D. Jiang, S. Sun, and Y. Yu, "Functional renyi differential privacy for generative modeling," in *Advances in Neural Information Processing Systems*, 2023.

[26] L. Xie, K. Lin, and et al., "Differentially private generative adversarial network," *CoRR*, vol. abs/1802.06739, 2018.

[27] T. Dockhorn, T. Cao, A. Vahdat, *et al.*, "Differentially private diffusion models," *Transactions on Machine Learning Research*, 2023.

[28] S. Ghalebikesabi, L. Berrada, S. Gowal, *et al.*, "Differentially private diffusion models generate useful synthetic images," *CoRR*, vol. abs/2302.13861, 2023.

[29] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography: Third Theory of Cryptography Conference,*, pp. 265–284, 2006.

[30] I. Mironov, K. Talwar, and L. Zhang, "Rényi differential privacy of the sampled gaussian mechanism," *CoRR*, vol. abs/1908.10530, 2019.

[31] B. Jähne, *Digital image processing*. Springer Science & Business Media, 2005.

[32] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems*, 2007.

[33] J. Zhang, M. Jalali, C. T. Li, and F. Farnia, "Unveiling differences in generative models: A scalable differential clustering approach," in *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 8269–8278, June 2025.

[34] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis and machine vision*. Springer, 2013.

[35] H. He, C. Bai, K. Xu, *et al.*, "Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[36] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, 2017.

[37] Z. Lin, V. Sekar, and G. Fanti, "Why spectral normalization stabilizes gans: Analysis and improvements," *Advances in neural information processing systems*, vol. 34, pp. 9625–9638, 2021.

[38] I. Mironov, "Renyi differential privacy," *CoRR*, vol. abs/1702.07476, 2017.

[39] C. Gong, K. Li, Z. Lin, and T. Wang, "From easy to hard++: Promoting differentially private image synthesis through spatial-frequency curriculum," *arXiv preprint arXiv:2601.06368*, 2026.

[40] S. Gopi, Y. T. Lee, and L. Wutschitz, "Numerical composition of differential privacy," in *Advances in Neural Information Processing Systems*, 2021.

[41] Y. Long, B. Wang, and et al., "G-PATE: scalable differentially private data generator via private aggregation of teacher discriminators," in *Advances in Neural Information Processing Systems*, 2021.

[42] Y. LeCun, L. Bottou, Y. Bengio, and et al., "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[43] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, 2017.

[44] A. Krizhevsky, "Learning multiple layers of features from tiny images," tech. rep., University of Toronto.

[45] P. Bándi, O. Geessink, Q. Manson, and et al., "From detection of individual metastases to classification of lymph node status at the patient level: The CAMELYON17 challenge," *IEEE Trans. Medical Imaging*, vol. 38, no. 2, pp. 550–560, 2019.

[46] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.

[47] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, "Improved precision and recall metric for assessing generative models," *Advances in neural information processing systems*, vol. 32, 2019.

[48] M. Jiralerspong, J. Bose, I. Gemp, *et al.*, "Feature likelihood divergence: evaluating the generalization of generative models using samples," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[49] S. Hod and R. Canetti, "Differentially private release of israel's national registry of live births," in *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 101–101, IEEE Computer Society, 2024.

[50] J. Deng, W. Dong, R. Socher, and et al., "Imagenet: A large-scale hierarchical image database," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, 2009.

[51] S. P. Liew, T. Takahashi, and M. Ueno, "PEARL: data synthesis via private embeddings and adversarial reconstruction learning," in *The Tenth International Conference on Learning Representations*, 2022.

[52] F. Harder, M. Jalali, D. J. Sutherland, and et al., "Pre-trained perceptual features improve differentially private image generation," *Trans. Mach. Learn. Res.*, 2023.

[53] J.-W. Chen, C.-M. Yu, C.-C. Kao, *et al.*, "Dpgen: Differentially private generative energy-guided network for natural image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8387–8396, 2022.

[54] Y. Yin, Z. Lin, M. Jin, G. Fanti, and V. Sekar, "Practical gan-based synthetic ip header trace generation using netshare," in *Proceedings of the ACM SIGCOMM 2022 Conference*, pp. 458–472, 2022.

[55] Z. Lin, A. Jain, C. Wang, *et al.*, "Using gans for sharing networked time series data: Challenges, initial promise, and open questions," in *Proceedings of the ACM internet measurement conference*, pp. 464–483, 2020.

[56] I. Escartín, "python avatars." https://github.com/ibonn/python_avatars, 2021.

[57] J. K. Christopher, S. Baek, and N. Fioretto, "Constrained synthesis with projected diffusion models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 89307–89333, 2024.

[58] Q. Liu, O. Deho, F. Vadiee, *et al.*, "Can synthetic data be fair and private? a comparative study of synthetic data generation and fairness algorithms," in *Proceedings of the 15th International Learning Analytics and Knowledge Conference*, pp. 591–600, 2025.

[59] B. An, M. Ding, T. Rabbani, *et al.*, "Waves: benchmarking the robustness of image watermarks," in *Proceedings of the 41st International Conference on Machine Learning*, pp. 1456–1492, 2024.

[60] A. Gretton, K. M. Borgwardt, M. J. Rasch, *et al.*, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, 2012.

[61] S. Arora, S. S. Du, W. Hu, *et al.*, "On exact computation with an infinitely wide neural net," *Advances in neural information processing systems*, vol. 32, 2019.

[62] D. Chen, T. Orekondy, and M. Fritz, "GS-WGAN: A gradient-sanitized approach for learning differentially private generators," in *Advances in Neural Information Processing Systems*, 2020.

[63] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, pp. 211–407, 2014.

[64] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *9th International Conference on Learning Representations, ICLR*, 2021.

[65] R. C. Gonzalez, *Digital image processing*. Pearson education india, 2009.

Due to space limitations, please refer to our full version [39] for additional appendices (Appendix C.3-F).

A Details of Rényi DP

This section outlines the computation of privacy costs using Rényi DP (RDP) [30], which offers a robust framework for tracking privacy loss in DP mechanisms.

Definition 2 (Sub-sampled Gaussian Mechanism (SGM [30]))**.** *Let $f : D_s \subseteq D \to \mathbb{R}^d$ be a function with sensitivity $\Delta_f = \max_{D\simeq D'} \|f(D) - f(D')\|_2$. Parameterized with a sampling rate $q \in (0,1]$ and noise standard deviation $\sigma > 0$, the SGM $\mathcal{Q}$ is defined as,*

$$\mathcal{Q}_{f,q,\sigma}(D) \triangleq f(S) + \mathcal{N}\left(0, \sigma^2 \Delta_f^2 \mathbb{I}\right)$$

where $S = \{x \,|\, x \in D$ selected independently with probability $q\}$ and $f(\varnothing) = 0$. The privacy loss of SGM can be tracked through Rényi DP [30], as elaborated in Definition 3. RDP can quantify the privacy loss of SGM accurately, as introduced in Theorem 4.

Definition 3 (*Rényi DP* [30])**.** *The Rényi divergence between two probability distributions Y and N is defined as, $D_\alpha(Y\|N) = \frac{1}{\alpha-1} \ln \mathbb{E}_{x\sim N}\left[\left(\frac{Y(x)}{N(x)}\right)^\alpha\right]$, where $\alpha > 1$ is a real number. A randomized mechanism $\mathcal{A}$ satisfies (α,γ)-RDP if, for any neighboring datasets D, D', and algorithm $\mathcal{Q}$, it holds that, $D_\alpha(\mathcal{Q}(D)\|\mathcal{Q}(D')) \le \gamma$.*

Given a batch size B, dataset size N, clipping hyperparameter C, and noise variance σ^2, as defined in Section 2.1, the sampling ratio is $q = B/N$. The RDP privacy cost for a single SGM $\mathcal{Q}_{f,q,\sigma}(D)$ is given by Theorem 4 [30].

Theorem 4. *Let $p_0 = \mathcal{N}(0, C^2\sigma^2)$ and $p_1 = \mathcal{N}(1, C^2\sigma^2)$ denote the probability density functions of two Gaussian distributions. A single SGM $\mathcal{Q}_{f,q,\sigma}(D)$ satisfies (α,γ_i)-RDP for any γ_i such that:*

$$\gamma_i \ge D_\alpha\left((1-q)p_0 + q p_1 \,\|\, p_0\right). \tag{1}$$

This theorem enables the calculation of the per-step privacy bound γ_i using the Rényi divergence. To express this in terms of (ε,δ)-DP, the following conversion applies.

Theorem 5 (*From (α,γ)-RDP to (ε,δ)-DP* [38])**.** *A mechanism $\mathcal{A}$ satisfying (α,γ)-RDP also satisfies (ε,δ)-DP for any $0 < \delta < 1$, where, $\varepsilon = \gamma + \frac{\ln(1/\delta)}{\alpha-1}$.*

Thus, by adjusting the noise variance σ^2, the final privacy cost $\varepsilon = \gamma + \frac{\ln(1/\delta)}{\alpha-1}$ can be tailored to meet a target privacy budget ε.

Privacy Composition. When applying multiple differentially private mechanisms sequentially, their privacy costs must be combined to determine the overall privacy guarantee. For RDP, the composition theorem states that if k mechanisms $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_k$ each satisfy (α,γ_i)-RDP, the composite mechanism satisfies (α,γ)-RDP, where $\gamma = \sum_{i=1}^k \gamma_i$.

This additive property simplifies privacy accounting across multiple steps, such as in DP-SGD iterations or multi-stage frameworks in FETA-Pro. The cumulative RDP cost (α,γ) can then be converted to (ε,δ)-DP using Theorem 5.

B Proof of Sensitivity

Proof of Theorem 1. *The query of frequency features of D_s has global sensitivity $\Delta_f = 1/N^*$. For any $\alpha > 1$, incorporating noise $\mathcal{N}\left(0, \sigma_f^2 \Delta_f^2 \mathbb{I}\right)$ into the mean random Fourier feature μ^p makes the query results satisfy (α,γ)-RDP for some γ.*

Proof. Let us assume two neighboring image datasets $D_s = \{h_i\}_{i=1}^N$ and $D_s' = \{h_i'\}_{i=1}^{N-1}$, where $h_i = h_i'$, $i \in [1, N-1]$. The sensitivity of the frequency features query is,

$$\begin{aligned}\Delta_f &= \max \left\| \frac{1}{N^*} \sum_{i=1}^{N-1} \phi(h_i) - \frac{1}{N^*} \sum_{i=1}^{N} \phi(h_i') \right\| \\ &= \max \left\| \frac{1}{N^*} \phi(h_N) \right\| \le \frac{1}{N^*}.\end{aligned}$$

To derive the third line from the second, we use the triangle inequality and the fact that $\|\phi(\cdot)\| = 1$. We refer to implementation in DPImageBench [4], using $N^* \approx N$. We ignore the privacy budget caused by the dataset size estimation.

Proof of Theorem 2. *The query of spatial feature h^{spat} has global sensitivity $\Delta_{spat} = C_t/B_t^*$. For any $\alpha > 1$, incorporating noise $\mathcal{N}\left(0, \sigma_t^2 \Delta_{spat}^2 \mathbb{I}\right)$ into the query result h^{spat} makes the query results satisfy (α,γ)-RDP for some γ.*

Proof. We provide the proof of global sensitivity as follows. For any two neighboring image subsets $D_s^{\text{sub}} = \{h_i\}_{i=1}^{B_t}$ and

$D_s^{\text{sub}'} = \{h_i\}_{i=1}^{B_t-1}$, the corresponding central images are h^{spat} and $h^{\text{spat}'}$, we have the following inequation,

$$\begin{aligned}\Delta_{\text{spat}} &= \left\| h^{\text{spat}} - h^{\text{spat}'} \right\|_2 \\ &= \left\| \frac{1}{B_t^*} \sum_{i=1}^{B_t} h_i - \frac{1}{B_t^*} \sum_{i=1}^{B_t-1} h_i \right\|_2 \\ &= \left\| \frac{1}{B_t^*} h_{B_t} \right\|_2 \leq \frac{C_t}{B_t^*}.\end{aligned}$$

Therefore, we have $\Delta_{\text{spat}} \leq \frac{C_t}{B_t^*}$.

Proof of Theorem 3. *Consider partitioning the dataset as $D_s = [D_s^1, D_s^2, \cdots, D_s^C]$, where the subsets are disjoint. By applying Equation (6) to each subset, we generate noisy frequency features. The combined features have a dimension of $K \cdot C$, where K is the predefined feature dimension and C is the number of subsets. Because the subsets are disjoint, this mechanism has the same (α, γ)-RDP guarantee as that in Theorem 1.*

Proof. we categorize D_s into C disjoint subsets, $D_s = [D_s^1, D_s^2, \cdots, D_s^C]$. Let us assume two neighboring image datasets $D_s = \{h_i\}_{i=1}^{N}$ and $D_s' = \{h_i'\}_{i=1}^{N-1}$, where $h_i = h_i'$, $i \in [1, N-1]$. Besides, we assume the different image $h_i \in D_s^C$. The sensitivity of the frequency features query is,

$$\begin{aligned}\Delta_f = \max \Bigg\| & \frac{1}{N^*} \left[\sum_{i=1}^{|D_s^1|} \phi(h_i), \cdots, \sum_{i=1}^{|D_s^C|} \phi(h_i) \right] \\ & - \frac{1}{N^*} \left[\sum_{i=1}^{|D_s^1|} \phi(h_i), \cdots, \sum_{i=1}^{|D_s^C|-1} \phi(h_i) \right] \Bigg\| \\ = \max & \left\| \frac{1}{N^*} [0, \cdots, \phi(h_N)] \right\| \leq \frac{1}{N^*}.\end{aligned}$$

The derivation from the second line to the third leverages the triangle inequality and the property that $\|\phi(\cdot)\| = 1$. Similar to the implementations in Theorem 1, we utilize the approximation $N^* \approx N$ and disregard the privacy budget incurred by this dataset size estimation.

C Implementation Details

This section introduces the baselines (including existing methods and variants of FETA-Pro), details of selected metrics, and hyper-parameter settings of FETA-Pro.

C.1 Beselines

In our experiments, we implement all baselines using the open-source DP image synthesis benchmark DPImageBench [4]. We introduce baselines as follows:

- **DP-MERF [19]:** DP-MERF leverages random feature representations of kernel mean embeddings with the Maximum Mean Discrepancy (MMD) [60] to minimize the distributional distance between real and synthetic data.
- **DP-NTK [24]:** DP-NTK uses Neural Tangent Kernels [61] to represent images, using the gradient of the neural network function as a feature map to extract perceptual features from original images for DP synthesis.
- **DP-Kernel [25]:** DP-Kernel uses functional RDP to privatize the loss function of the data generator within a reproducing kernel Hilbert space, enabling DP image synthesis.
- **GS-WGAN [62]:** GS-WGAN perturbs only the discriminator's feedback to guide the generator, ensuring DP. By applying the chain rule, it decomposes the generator's gradient into an *upstream gradient* (the discriminator's output with respect to the generated image) and a *local gradient* (the generated image with respect to the generator's parameters), perturbing only the upstream gradient for DP, as the discriminator solely accesses sensitive images.
- **DP-GAN [26]:** DP-GAN trains the discriminator network on sensitive images using DP-SGD [14], ensuring that the discriminator weights satisfy differential privacy (DP). The generator's weight updates depend solely on the discriminator. By the post-processing property of DP [63], the generator also satisfies DP.
- **DPDM [27]:** DPDM trains diffusion models on sensitive images using DP-SGD [14]. It introduces noise multiplicity, a modification to DP-SGD, to mitigate the adverse effects of injected noise on gradients.
- **DP-FETA [15]:** DP-FETA uses a two-stage training process: (1) It extracts basic features (e.g., outlines, colors) from sensitive images using 'central images' – derived from central tendency measures (e.g., mean, mode). (2) Fine-tuning on sensitive images with DP-SGD [14].

Then, we introduce the variants of FETA-Pro studied in Section 5.2 as follows.

- **FETA-Pro-No-Auxiliary**. This method means using the synthesizer, diffusion model e_θ, to learn the frequency feature directly. Similar to Equation (8), at each training iteration, this method uses the diffusion model to generate a batch of synthetic images, and then it optimizes the model parameters to minimize the difference between the frequency features of synthetic images and sensitive images. The formal objective is

$$\mathcal{L}(\theta) = \|\tilde{\mu} - \mu^s(\text{Sampler}(e_\theta, z))\|_2, \tag{2}$$

where z is a batch of random noise. 'Sampler' uses e_θ to denoise z into a batch of synthetic images through multiple steps of denoising. Although we can obtain the denoised images before the final step, these images are still too noisy

Table 9: Hyper-parameter settings of FETA-Pro. 'LR' denotes Learning Rate.

Hyper-parameter	$\varepsilon = 1.0$					$\varepsilon = 10.0$				
	MNIST	F-MNIST	CIFAR-10	CelebA	Camelyon	MNIST	F-MNIST	CIFAR-10	CelebA	Camelyon
Noise scale σ_f	26.6	26.6	32.6	25.0	49.6	7.4	7.4	8.2	8.2	8.4
Noise scale σ_t	20	20	15	15	10	5	5	5	50	5
Spatial domain Epoch	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000	1,000
Spatial domain LR	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$
Spatial domain Batch size	50	50	50	50	50	50	50	50	50	50
Spatial domain norm C_t	28	28	55	55	55	28	28	55	55	55
Spatial sample rate q_t	0.11	0.11	0.13	0.08	0.04	0.11	0.11	0.13	0.08	0.04
Spatial image amount N_t	50	50	50	500	500	50	50	50	500	500
Frequency domain Epoch	10	10	10	10	10	10	10	10	10	10
Frequency domain LR	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$
Auxiliary Generator Epoch M	5	5	5	5	5	5	5	5	5	5
Auxiliary Generator LR	0.01	0.01	0.01	0.005	0.005	0.01	0.01	0.005	0.01	0.005
Auxiliary Generator Batch size	100	100	100	500	500	100	100	500	100	500
Frequency domain Batch size B_f	256	256	256	256	256	256	256	256	256	256
Frequency feature dimension K	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000
Size of D_s^f Image Dataset N_f	60,000	60,000	60,000	60,000	60,000	60,000	60,000	60,000	60,000	60,000
Fine-tuning Epoch	150	150	150	150	50	150	150	150	150	50
Fine-tuning LR	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$	$3e^{-4}$
Fine-tuning Batch size	4096	4096	4096	4096	4096	4096	4096	4096	4096	4096
Fine-tuning grad. norm	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
Fine-tuning sample rate	$7.4e^{-2}$	$7.4e^{-2}$	$9.1e^{-2}$	$2.8e^{-2}$	$1.5e^{-2}$	$7.4e^{-2}$	$7.4e^{-2}$	$9.1e^{-2}$	$2.8e^{-2}$	$1.5e^{-2}$

and can hardly contain useful frequency features. Therefore, we use the clean image at the final step to calculate its frequency feature. This paper uses the DDIM Sampler [64] with a sampling step of 50.

- **FETA-Pro-DM-Auxiliary**. This method means using diffusion models as the auxiliary generator, and other components keep consistent with those in FETA-Pro.

C.2 Dataset Details

In this paper, we conduct experiments on five image datasets that are widely used in prior works [4, 5, 7, 9, 10, 15] to verify the effectiveness of FETA-Procompared with baselines: `MNIST`, `FashionMNIST` [43] (`F-MNIST`), `CIFAR-10` [44], `CelebA` [22], and `Camelyon` [45].

The `MNIST` dataset consists of 70,000 grayscale images of handwritten digits (0–9). `F-MNIST` includes 70,000 images of 10 distinct fashion products. `CIFAR-10` comprises 10 classes of natural images, consisting of 60,000 images. Compared to `MNIST`, `F-MNIST`, and `CIFAR-10`, `CelebA` and `Camelyon` are more sensitive datasets. `CelebA` contains over 202,599 facial images of 10,177 celebrities, each annotated with 40 attributes; following prior work [4, 6, 15], we use the 'Gender' attribute to classify images as male or female. `Camelyon` includes 455,954 histopathological image patches of human tissue, labeled based on the presence of at least one tumor cell pixel. As shown in Table 3, all datasets are split into training, validation, and test sets. For `CelebA` and `Camelyon`, all images are center-cropped and resized to 32×32 pixels.

C.3 Details of Metrics

We use 'entropy' and 'texture complexity' to evaluate the complexity of images. We elaborate on them as follows.

- **Entropy:** Entropy measures the level of uncertainty in a set of data. A high entropy value means the pixel values are highly random and diverse, indicating a greater amount of information and, therefore, more complexity [34]. Calculating image entropy involves these steps: (1) *Grayscale Conversion*: If the image is in color, we first convert it to grayscale so that each pixel has a single value (0-255). (2) *Histogram:* We create a histogram of the image's pixel values, which gives a probability distribution p_k. Here, p_k is the number of pixels with a value of k. *Shannon Entropy:* We then use the Shannon Entropy [34] to calculate the entropy value: $H = -\sum_{k=0}^{K-1} p_j \log(p_k)$.

- **Texture complexity:** Texture complexity evaluates an image based on the richness of its visual details, patterns, and structures. It captures the kind of complexity the human visual system perceives, such as the number of edges and the regularity of patterns. An image with more details and irregular patterns has a higher texture complexity [65].

We evaluate the fidelity and utility of the synthetic dataset using two metrics: Fr´echet Inception Distance (FID) and downstream classification accuracy (Acc). We generate 60,000 synthetic images for evaluations. We implement these metrics by referring to the adoptions in DPImageBench [4].