SCALABLE DIFFERENTIALLY PRIVATE COMPRESSION VIA DIFFUSION AND STOCHASTIC CODES

Anonymous authors
Paper under double-blind review

ABSTRACT

Differential privacy (DP) provides formal guarantees for releasing personal data. However, releasing high-dimensional data, such as images, is costly: privatization is typically achieved by adding noise, and a compression codec applied after privatization spends most of its bits compressing noise.

We address this challenge with *DP-DiPP*, which jointly privatizes and compresses an image by combining stochastic codes with diffusion models. DP-DiPP is highly flexible: the practitioner has direct control over the compression rate-privacy-utility tradeoff. We develop a step-limited variant of the the Poisson private representation (PPR) of (Liu et al., 2024) to encode the outputs of privacy mechanisms. We then combine it with DiffC, a diffusion-based lossy compression method, to obtain a scalable image compressor with DP. Our experiments on privatized image classification demonstrate that DP-DiPP significantly outperforms the baseline, achieving $\times$10-30 higher compression on CIFAR-10 and $\times$180-400 on the Shenzhen chest X-ray set (Jaeger et al., 2014), a high-resolution medical image dataset, at matched utility.

1 INTRODUCTION

Differential privacy (DP) (Dwork et al., 2006; Dwork & Roth, 2014) has become the standard method to provide formal privacy guarantees in sensitive data release. It has been widely deployed, e.g., for collecting user data (Erlingsson et al., 2014), releasing privatized tabular data, like census data (United States Census Bureau, 2021), and preventing generative models from memorizing training samples (Liu et al., 2023). In this paper, we adopt *local differential privacy* (LDP) as the privacy framework, a stronger variant of DP. LDP protects individuals' data by requiring that a system's output reveals only limited information about its input. In practice, this idea translates to randomizing the sensitive data X before revealing it to an untrusted party, according to a *privacy mechanism*. For example, if X represents location data, one possible privacy mechanism is to add appropriately calibrated noise η to prevent precise localization, that is, $Y = X + \eta$. A natural trade-off arises between the privatized data's *utility* for its downstream task and its privacy guarantee.

Scaling LDP to high-dimensional data such as images, audio, or video, adds a third dimension to this trade-off, the bit cost of storing or transmitting privatized images. The noise added for privacy guarantees can destroy the structure exploited by compression codecs, requiring full bit depth for lossless compression. Hence, we study privatized compression of high-resolution images, where privacy, downstream utility and compression rate must be controlled jointly.

Our approach builds upon the emerging field of data compression with stochastic codes (Flamich, 2024; Li et al., 2024; Flamich & Gündüz, 2026). In practice, stochastic codes serve as a *randomized lossy compression mechanism*: consider an input X (modeled as a random variable) and a perturbed, lossy version $Y = X + \eta$, where η is independent of X. Then, for a given X, stochastic codes produce a bitstring S that encodes one sample of $Y \mid X$, with the expected length of S as small as possible. By utilizing stochastic codes, the generated sample exactly follows the desired conditional distribution. This suggests applying them to compressing the output of privacy mechanisms, where preserving the exact sample distribution preserves other desirable statistical properties of the original privacy mechanism, such as unbiasedness and Gaussianity. However, unlike in conventional compression codecs, the bitstream produced by stochastic codes is not a post-processing of the privacy mechanism, so the privacy guarantee does not carry over automatically (Section 2.2). Poisson private

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

representation (PPR; Liu et al., 2024) is a stochastic code that *does* provide LDP guarantees. PPR can encode any pure ϵ-LDP mechanism, ensuring that its output is at least $2\alpha\epsilon$-LDP, where $\alpha > 1$ is a tunable parameter that interpolates between the code's privacy and the expected codelength.

Applying PPR to complex data is challenging due to its slow runtime, scaling super-exponentially with the information content. We introduce an approximate version of PPR, called *step-limited PPR*, which provides the same privacy guarantee as PPR with a fixed runtime. We then combine our step-limited PPR code with DiffC, a variable-rate, diffusion-based compression framework (Theis et al., 2022). DiffC progressively encodes each denoising step of a Gaussian diffusion model, thereby decomposing the compression problem into smaller, manageable steps. However, additive Gaussian noise does not yield a pure ϵ-LDP mechanism. Hence, we approximate the Gaussian denoising targets at each step with moment-matched Laplace distributions, which do provide pure ϵ-LDP guarantees. Somewhat surprisingly, we empirically find that this approximation does not degrade the denoiser's performance, while strengthening its formal privacy guarantee. We call the resulting method *Differentially Private DiffC with Poisson Processes* (DP-DiPP, pronounced *dippy-dip*).

DP-DiPP jointly compresses and privatizes the data, rather than privatizing it first and then compressing the privatized data. We evaluate DP-DiPP on CIFAR-10 and high-resolution chest X-ray images (Jaeger et al., 2014), and compare them with baseline methods including the recently proposed LDP-Slicing (Cao et al., 2026). We find that DP-DiPP performs significantly better.

In summary, our main contributions are as follows:

- In Section 3, we introduce step-limited PPR, a fixed-runtime parallelizable variant of PPR Liu et al. (2024), which yields a $2\alpha\epsilon$-LDP certificate for any ϵ-LDP mechanism, where $\alpha \geq 1$ is a tunable parameter.
- In Section 4, we modify DiffC (Theis et al., 2022), a variable-rate, Gaussian diffusion-based lossy data compressor, by replacing the Gaussian targets with Laplace distributions, which turns the reverse process into an ϵ-LDP mechanism. We show empirically that this approximation does not significantly degrade the denoiser's performance.
- In Section 5, we apply the step-limited PPR mechanism to the modified DiffC to obtain *DP-DiPP*, and validate its performance on privatized image compression tasks on different datasets. We show that DP-DiPP consistently reduces bitrate by a factor of $\times$10-30 for CIFAR-10 and $\times$180-400 for Shenzhen CXR while maintaining matched utility (as measured by classification accuracy).

2 Background

2.1 Privacy Mechanisms and DP

DP provides formal certificates that quantify how much information a system reveals about its input (Dwork et al., 2006; Dwork & Roth, 2014). We concern ourselves with its local variant, LDP: we randomize a sensitive input x using a *privacy mechanism* $M(x)$ to reduce the information leaked about x. However, not any randomization will do: in ε-LDP we choose a privacy parameter $\varepsilon \geq 0$ and require the mechanism M to satisfy, for any two inputs x, x' and any measurable event S, $\mathbb{P}[M(x) \in S] \leq e^{\varepsilon}\mathbb{P}[M(x') \in S]$. It can be interpreted as ensuring a certain level of difficulty in distinguishing whether the output of M was generated from input x or x'.

Naturally, we wish to use privatized data for some purpose. We measure the usefulness of the released data with a *utility function* u, and maximize the expected utility $\mathbb{E}[u(M(X))]$ for a given privacy level ϵ. In most cases of practical interest, maintaining privacy comes at the cost of utility; hence, we seek to characterize the system's *utility-privacy tradeoff* (Duchi et al., 2013; Chen et al., 2020).

2.2 Compressing DP Mechanisms

When the sensitive data is scalar-valued or low-dimensional, it is usually feasible to store or transmit it as-is, uncompressed. However, once the size or dimensionality of the data exceeds a reasonable threshold, such as for image, audio, or video data, compression becomes necessary.

The straightforward solution is to apply the mechanism M first and then losslessly compress the privatized output using, e.g., `zip` for general data or `PNG` for images. We consider lossless compression only, as using a lossy compressor could further compromise the utility of the privatized data in ways that are difficult to control. Denoting a lossless compressor as C, the privatized and then compressed version of data x is $C(M(x))$. By the *post processing* property of DP mechanisms (Dwork & Roth, 2014), if M is ε-LDP, then so is $C(M(x))$.

However, since privacy mechanisms randomize the data, the privatized output has higher entropy, and the code must describe the noise. In the case of a perfectly privatized image, all the structure is lost, and losslessly compressing each purely random pixel requires the full bit depth. If the privacy mechanisms rely on continuous noise, such as Gaussian or Laplace noise, it is impossible to compress the privatized data losslessly; see (Shahmiri et al., 2024) for a resolution of this issue for Laplace noise. This is referred to as the "communication-privacy-accuracy trilemma" (Chen et al., 2020; Sreekumar & Gündüz, 2019).

The inefficiency of the privatize-then-compress approach stems from its sequential structure: the compressor only ever handles randomized data. In contrast, using stochastic codes allows us to *jointly privatize and compress*, which avoids compressing the noise. Formally, given a conditional distribution $P_{M|X}$, a stochastic code is a triplet $(Z, \mathrm{enc}, \mathrm{dec})$ made up of:

- A random variable Z independent of the input X, shared between the sender/compressor and the receiver/decompressor, and hence called the *common randomness*. In practice, Z can be realized by using a shared pseudo-random number generator (PRNG) and a common seed for the PRNG.
- A function $\mathrm{enc}(x, z)$ mapping an input x and a realization of Z to a finite-length bitstring, called the *encoder*.
- A function $\mathrm{dec}(s, z)$ mapping a bitstring s and a realization of Z to the support of M, called the *decoder*.

Finally, to qualify as a stochastic code for $P_{M|X}$, the triplet must satisfy, for all x,

$$\mathrm{dec}(\mathrm{enc}(x, Z), Z) \sim P_{M|X=x} \tag{1}$$

Note that the randomness arises purely from the synchronized random variable Z. Intuitively, one can think of $s = \mathrm{enc}(x, Z)$ as encoding a single random sample from $P_{M|X}$. Somewhat surprisingly, it can be shown that not only can we construct a stochastic code for virtually any conditional distribution $P_{M|X}$, but we can do so with an expected codelength $\mathbb{E}[|\mathrm{enc}(X, Z)|]$ close to $I[M\ ;\ X]$, which is optimal (Li & El Gamal, 2018; Li et al., 2024; Flamich, 2024).

As we can construct a stochastic code to compress a sample from an arbitrary conditional distribution, and privacy mechanisms may be conceived as conditional distributions, it is a natural next step to combine the two. However, things become more involved regarding the privacy certificate: with stochastic codes, we have the Markov chain

$$X \to \mathrm{enc}(X, Z) \to \underbrace{\mathrm{dec}(\mathrm{enc}(X, Z), Z)}_{\sim M(X)} \tag{2}$$

as opposed to $X \to M(X) \to C(M(X))$ as we had with the privatize-then-compress solution. This observation shows that even if $M(X)$ is ε-LDP, post processing does not apply and $\mathrm{enc}(X, Z)$ is not immediately ε-LDP. Instead, we need to take special care in constructing $(Z, \mathrm{enc}, \mathrm{dec})$ so that it can retain (some of) the privacy guarantees of the mechanism M that it is compressing.

The first works to construct such stochastic codes were Feldman & Talwar (2021); Triastcyn et al. (2021); Hasırcıoğlu & Gündüz (2024) and Shah et al. (2022), however, they have suboptimal compression-privacy performance. Instead, we base our work on the Poisson private representation (PPR; Liu et al., 2024), the only known stochastic code that exactly simulates arbitrary differential privacy mechanisms with optimal expected codelength.

2.3 The Poisson Private Representation (PPR)

As we outlined above, to define the PPR code for an ϵ-LDP mechanism M implementing the conditional distribution $P_{M|X}$, we need to specify the triplet $(Z, \mathrm{enc}, \mathrm{dec})$ that satisfies Equation (1).

To begin, we assume that there is a marginal distribution Q_M over the support of M such that $P_{M|X=x} \ll Q_M$, i.e., the support of $P_{M|X=x}$ is contained in the support of Q_M for all x. Furthermore, we assume we can easily generate samples from Q_M and that we can compute the density ratio (technically, the Radon-Nikodym derivative) $\frac{dP_{M|X}}{dQ_M}(m \mid x)$. In our application to diffusion models, Q_M will be either a Gaussian or Laplace distribution independent of X while $P_{M|X}$ will be a Gaussian or Laplace distribution that does depend on X, and the above conditions will be satisfied.

Definition 1 (PPR (Liu et al., 2024))**.** Let $P_{M|X}$ be an ϵ-LDP mechanism and Q_M be a probability distribution such that its support contains the support of $P_{M|X=x}$ for all x. For convenience, let $r(m \mid x) = \frac{dP_{M|X}}{dQ_M}(m \mid x)$. Let $\alpha > 1$, we define the PPR code as follows:

- We set the common randomness as an infinite sequence of pairs of random variables $Z \leftarrow \{(M_k, T_k)\}_{k=1}^{\infty}$, where $M_k \sim Q_M$ and $(T_k - T_{k-1}) \sim \mathrm{Exp}(1)$ with the convention that $T_0 = 0$. Such an infinite sequence is called a *Poisson process*, the namesake of PPR.
- We define the encoder in two steps. First, using random variables $V = \{V_k\}_{k=1}^{\infty}$ where $V_k \sim \mathrm{Exp}(1)$ are not known to the decoder, the encoder computes
$$K(X, Z, V) = \arg\min_{k \in \mathbb{N}} V_k \cdot T_k^{\alpha} \cdot r(M_k \mid X)^{-\alpha} \tag{3}$$
Although the minimization is over an infinite set, it is nonetheless possible to find a minimizer in finite time. As K is an integer, we need to map it to a bitstring to get a valid stochastic code. To this end, we construct an Shannon-Fano code $\zeta(k)$ from a zeta distribution, so that setting $\mathrm{enc}(X, Z, V) = \zeta(K(X, Z, V))$ and $\ell = D_{\mathrm{KL}}[P_{M|X} \parallel Q_M] + \log(3.56)/\min\{(\alpha - 1)/2, 1\}$,
$$\mathbb{E}[|\mathrm{enc}(X, Z, V)|] \leq \ell + \log(\ell + 1) + 2 \tag{4}$$
For the precise encoding details, see Liu et al. (2024).
- Finally, the decoder simply recovers the selected index K from the ζ-code and then selects the sample M_K that the encoder computed from $Z = \{(M_i, T_i)\}_{i=1}^{\infty}$:
$$\mathrm{dec}(S, Z) \leftarrow M_{\zeta^{-1}(S)} \tag{5}$$

The rationale for the above definition of the PPR code is threefold: first, one can show that given our selection rule in Equation (3), we get $M_{K(x,Z,V)} \sim P_{M|X=x}$, and hence $\mathrm{dec}(\mathrm{enc}(x, Z), Z) \sim P_{M|X=x}$ as required. Second, the expected codelength (Equation (4)) for any fixed α is close to optimal (Liu et al., 2024; Li et al., 2024). Third, Theorem 4.5 of Liu et al. (2024) shows that when the mechanism $M(x)$ is ϵ-LDP, then $(Z, \mathrm{enc}(x, Z, V))$ is $2\alpha\epsilon$-LDP, and hence, so is $\mathrm{dec}(\mathrm{enc}(X, Z), Z)$ by the post processing property.

Finally, we must deal with the runtime $N(X, Z, V)$ of PPR. It can be shown that PPR belongs to the class of *selection samplers* (Flamich & Wells, 2024), and thus, unfortunately, its expected runtime upon input x is at least $\mathbb{E}[N(x, Z, V)] \geq \|r(\cdot \mid x)\|_{\infty}$. This is especially problematic for our intended application to diffusion models, in which case the expected runtime can be infinite since the density ratios involved can be unbounded! Thus, as our first contribution, in Section 3, we shall develop a variant of PPR that can compress privacy mechanisms with unbounded density ratios at the cost of encoding them only approximately. However, as our Theorem 3.1 shows, the error of approximate PPR's output is small in total variation, and the algorithm is guaranteed to have a short codelength and retain its desirable privacy certificate, while having a fixed runtime, with the possibility to parallelize the most expensive computations to further improve the runtime.

Unfortunately, for the approximation guarantee to hold, the runtime has to be $\mathcal{O}(\exp(D_{\mathrm{KL}}[P_{M|X} \parallel Q_M]))$, which is prohibitive when the KL is thousands of bits, e.g., for the whole image. Thus, finally, we describe DiffC, a framework that enables scaling stochastic codes to high-resolution data by combining them with diffusion models.

2.4 Data Compression with Diffusion Models

A denoising diffusion probabilistic model (DDPM; Ho et al., 2020; Sohl-Dickstein et al., 2015) consists of a fixed forward noising process and a learned reverse denoising process. The forward

process gradually corrupts a clean sample $x_0 \sim q(x_0)$ with Gaussian noise so that, at any time-step $t \in [0, T]$, the marginal admits the closed form

$$x_t = \gamma_t x_0 + \sigma_t \eta_t, \quad \eta_t \sim \mathcal{N}(0, \mathrm{I}),$$

where $\gamma_t = \sqrt{1 - \sigma_t^2}$ and σ_t^2 is a monotonically increasing *noise schedule* with $\sigma_0 \approx 0$ and $\sigma_T \approx 1$. We write $\mathrm{SNR}(t) = \gamma_t^2/\sigma_t^2$ for the signal-to-noise ratio at time-step t. The reverse process is parametrized as a Markov chain $p_\theta(x_s \mid x_t)$ trained to minimize the per-step KL divergence $D_{\mathrm{KL}}[q(x_s \mid x_t) \parallel p_\theta(x_s \mid x_t)]$ for $s < t$. Note that this objective is equivalent to minimizing $D_{\mathrm{KL}}[q(x_s \mid x_t, x_0) \parallel p_\theta(x_s \mid x_t)]$, where conditioning on x_0 yields a closed-form Gaussian posterior

$$q(x_s \mid x_t, x_0) = \mathcal{N}(\mu_q(x_t, x_0; s, t), \sigma^2(s, t)\mathrm{I}) \tag{6}$$

$$\mu_q(x_t, x_0; s, t) = \frac{\gamma_s \sigma_{t|s}^2}{\sigma_t^2} x_0 + \frac{\gamma_{t|s} \sigma_s^2}{\sigma_t^2} x_t \tag{7}$$

where $\gamma_{t|s} = \frac{\gamma_t}{\gamma_s}$, $\sigma_{t|s}^2 = \sigma_t^2 - \gamma_{t|s}^2 \sigma_s^2$, and $\sigma(s, t) = \frac{\sigma_{t|s}\sigma_s}{\sigma_t}$.

Theis et al. (2022) observed that the forward and reverse processes together admit a natural progressive lossy compression interpretation: We can use a stochastic code to encode a sample x_s from $q(x_s \mid x_t, x_0)$ using the model proposal $p_\theta(x_s \mid x_t)$, terminating at any intermediate time-step to trade off bitrate for distortion. Theis et al. (2022) called this framework ***DiffC*** and analyzed its rate-distortion performance assuming an ideal stochastic code. Vonderfecht & Liu (2025) proposed the first practical implementation using latent diffusion models, including Stable Diffusion 1.5/2.1/XL and FLUX.1-dev (Esser et al., 2024). We took the implementation of Vonderfecht & Liu (2025) as the basis for our work, see Appendix B for a more detailed description of their approach. We also note, that although the FLUX.1-dev is a rectified flow model, stochastic sampling along its probability flow can be shown to be equivalent to DDPM sampling Lai et al. (2025), and hence we may use it for DiffC. Flamich & Gündüz (2026) recently revisited path-based encoding in DiffC from a stochastic-coding perspective. They observed that under the assumption of a perfect denoiser, the KL divergence between joint forward and reverse paths on any common discretization $s = t_0 < t_1 < \cdots < t_n = T$ equals the KL divergence between their endpoint marginals at s. This observation means that encoding the final denoising step and the entire denoising path has roughly the same compression cost using stochastic codes. However, encoding the path is practically feasible.

3 Approximate PPR

As discussed in Section 2.2, PPR's exactness can be a drawback: its runtime scales as $\left\| \frac{dP_{M|X}}{dQ_M} \right\|_\infty$, which can be infinite even when $D_{\mathrm{KL}}[P_{M|X} \parallel Q_M]$ is finite. We address this issue by developing an approximate variant of PPR whose output is no longer an exact sample from $P_{M|X}$ but is still suitably close, as measured by the total variation distance. Furthermore, this variant will retain its close-to-optimal expected codelength and its average runtime scales as $\mathcal{O}(\exp(D_{\mathrm{KL}}[P_{M|X} \parallel Q_M]))$.

To achieve our goal, we adopt the deceptively simple strategy of Flamich & Wells (2024): we introduce a budget $n \in \mathbb{N}$ for the maximum number of samples PPR is allowed to examine. If the algorithm exceeds this budget, we terminate it and return the best index found so far.

Definition 2 (Step-limited PPR)**.** Let $P_{M|X}, Q_M, r(m \mid x), Z = \{(M_i, T_i)\}_{i=1}^\infty, V = \{V_i\}_{i=1}^\infty$ and ζ the Shannon-Fano code constructed from a zeta distribution as in Definition 1. In particular, recall that $P_{M|X}$ is an ϵ-LDP mechanism whose output we wish to encode. Now, let $n \in \mathbb{N}$ and $\alpha > 1$. We define the step-limited PPR algorithm as follows:

1. The common randomness is $Z = \{(M_i, T_i)\}_{i=1}^\infty$.
2. For the encoder, we define the step-limited index

$$K_n^\alpha(x, Z, V) = \operatorname*{arg\,min}_{k \in [1:n]} \left\{ V_k \left(\frac{T_k}{r(M_k \mid x)} \right)^\alpha \right\} \tag{8}$$

   Then, we set the encoder as

$$\mathrm{enc}_n^\alpha(x, Z, V) = \zeta(K_n^\alpha(x, Z, V)) \tag{9}$$

3. The decoder is the same as in Definition 1:

$$\mathrm{dec}(s, Z) = M_{\zeta^{-1}(s)} \tag{10}$$

By definition, the runtime of step-limited PPR is finite. But how should we choose the budget n? Besides proving that our algorithm retains PPR's LDP certificate and close-to-optimal codelength, the following result shows that choosing $n = \mathcal{O}(\exp(D_{\mathrm{KL}}[P_{M|X} \parallel Q_M]))$ ensures the algorithm's output distribution will be close to the original mechanism. The proof is in App. C.

Theorem 3.1. *Let $P_{M|X}, Q_M, K_n^\alpha, V$ and let $(Z, \mathrm{enc}, \mathrm{dec})$ be the step-limited PPR code with budget n and $\alpha > 1$ as in Definition 2. Take $\ell_\alpha(X) = D_{\mathrm{KL}}[P_{M|X} \parallel Q_M] + \frac{\log 3.56}{\min\{(\alpha-1)/2, 1\}}$. Then,*

1. ***LDP guarantee:*** *$\mathrm{enc}_n^\alpha(x, Z, V)$ is $2\alpha\epsilon$-LDP.*
2. ***Codelength:*** *Akin to exact PPR, we have for all x:*

$$\mathbb{E}[|\mathrm{enc}_n^\alpha(x, Z, V)|] \le \ell_\alpha(x) + \log(\ell_\alpha(x) + 1) + 2$$

3. ***Approximation quality:*** *Let $\beta > 0$ and let $Q_{M|X}^{(n)}$ denote the distribution of $\mathrm{dec}(\mathrm{enc}_n^\alpha(x, Z, V), Z)$. Then, choosing $n \ge 2^{\ell_\alpha(X)/\beta}$ guarantees that*

$$D_{\mathrm{TV}}[Q_{M|X=x}^{(n)} \parallel P_{M|X=x}] \le \beta \tag{11}$$

4 Privatized Data Compression with Diffusion

4.1 The Denoising Process as a Privacy Mechanism

Our key observation is that each reverse step of DDPM, conditioned on the original image x_0, is an additive-noise mechanism for x_0 with a closed-form budget (see Appendix D):

$$M(x_0) = \mu_q(x_t, x_0; s, t) + \sigma(s, t)\, \eta, \quad \eta \sim \mathcal{N}(0, \mathrm{I}) \tag{12}$$

Unfortunately, the Gaussian mechanism does not yield a *pure* ϵ-LDP certificate, and hence our Theorem 3.1 does not apply if we apply step-limited PPR to it. To circumvent this issue, we moment-match Laplace distributions to the Gaussian targets and proposals at each time step and use those for denoising and compression instead. Thus, the privacy mechanism becomes

$$M(x_0) = \mu_q(x_t, x_0; s, t) + b(s, t)\, \eta, \quad \eta \sim \mathcal{L}(0, \mathrm{I}) \tag{13}$$

where $b(s, t) = \sigma(s, t)/\sqrt{2}$. For a discussion of high-dimensional Laplace distribution on differential privacy, see (Andrés et al., 2013).

In what follows, we aim to provide per-pixel guarantees for image data. Thus, $x_0^{(i)}$ will denote a pixel on an image, consisting of C channels (e.g., $C = 3$ for RGB), with $x_0^{(i,c)}$ denoting channel c. We start with per-subpixel guarantees, which we also refer to as "per-channel," from which we recover the per-pixel guarantees. Following the steps in Dwork & Roth (2014, Theorem 3.6) applied to a single channel, the per-channel mechanism satisfies ϵ_{ch}-DP where

$$\epsilon_{\mathrm{ch}} = \frac{\gamma_s \sigma_{t|s}^2}{\sigma_t^2 b(s, t)} \cdot \sup |x_0^{(i,c)} - x_0'^{(i,c)}| \tag{14}$$

and the supremum is taken over all possible images. This admits the nicer form (see Appendix D):

$$\epsilon_{\mathrm{ch}} = \sqrt{2(\mathrm{SNR}(s) - \mathrm{SNR}(t))}\, \sup |x_0^{(i,c)} - x_0'^{(i,c)}|$$

By basic composition over the C channels, the per-pixel $\epsilon_{t\to s}$-DP guarantee per step is $\epsilon_{t\to s} = C\, \epsilon_{\mathrm{ch}}$. In all experiments, pixels are scaled to $[-1, 1]$, so $\sup |x_0^{(i,c)} - x_0'^{(i,c)}| = 2$, and for the single-channel X-rays the gray plane is copied to $C = 3$ channels before diffusion, so the per-pixel certificate is $3\epsilon_{\mathrm{ch}}$.

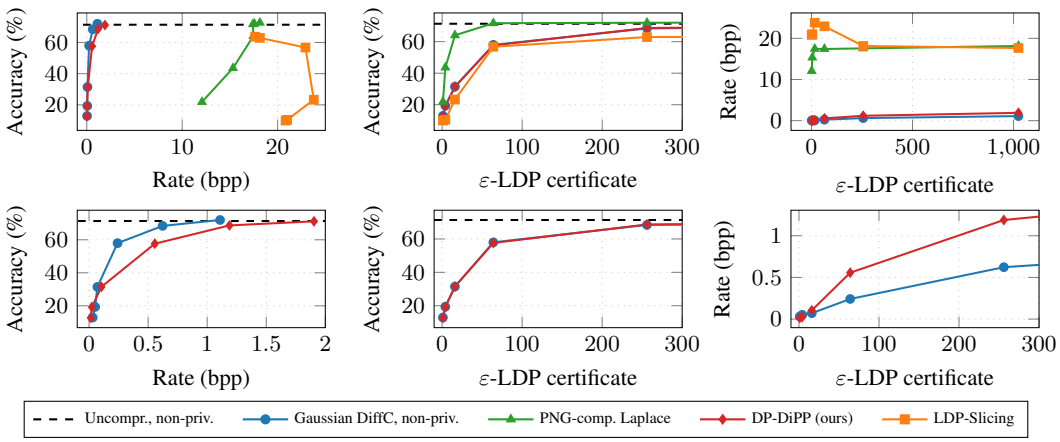

Figure 1: Experimental results on training an image classifier from a dataset of privatized and compressed CIFAR-10 images. Note that Gaussian DiffC **does not** have an LDP certificate and is included only for illustration. The left and middle columns also include the classification accuracy from the original, unprivatized data for comparison. In each case, the markers are connected by lines **in order of increasing** ϵ, where ϵ is the method's LDP guarantee. **Left column:** The rate-utility tradeoff, as measured by bits per pixel (bpp) and test classification accuracy, respectively. **Middle column:** Privacy-utility tradeoff, where the method's LDP certificate measures privacy. **Right column:** Rate-privacy tradeoff. **Top row:** comparisons among 1) the privatize-then-compress method using 2) Gaussian DiffC 3) LDP-Slicing and 4) DP-DiPP. **Bottom row:** Same comparisons as top row with the privatize-then-compress method omitted.

4.2 Compressing the Privacy Mechanism

In DiffC (Theis et al., 2022), the output of each denoising step is not released directly but compressed via a stochastic code; here, we apply the PPR code. Under PPR, at each step we compress the privacy mechanism $M_{t\to s}(x_0)$, and the compressed output inherits a privacy guarantee that depends on PPR's parameter α and the privacy parameters of the mechanism being compressed. Note that in Vonderfecht & Liu (2025), the code they used is equivalent to our PPR code with $\alpha = \infty$, which achieves maximal compression efficiency but provides no privacy guarantee.

However, as we explained in Section 2.3, exact PPR (i.e., the budget is $n = \infty$) requires an upper bound on the density ratio between the target and proposal distributions. In the Gaussian case of Equation (12), both distributions are Gaussians with the same variance, so the density ratio is unbounded. Fortunately, with our moment-matched Laplace distributions (Equation (13)) this is no longer an issue: the density ratio of two Laplace distributions with the same scale is always bounded. However, step-limited PPR still provides a significant computational advantage over its exact variant. Since we choose the step-budget n ahead of time, we can draw the n proposal samples and evaluate their density ratios in parallel on a GPU using a custom CUDA kernel (see Appendix E).

Thus, we use step-limited PPR (Definition 2). Combined with the per-step, per-pixel guarantee in Section 4.1, the PPR code satisfies a $2\alpha\epsilon_{t\to s}$-LDP per-pixel guarantee at each step. By sequential composition, the total per-pixel privacy certificate across n encoding steps is $\epsilon_{\text{total}} = 2\alpha \sum_{i=1}^{n} \epsilon_{t_i \to t_{i-1}}$.

We show experimentally that, for compression to the same final time-step, replacing the Gaussian channel with the moment-matched Laplace channel and $\alpha = \infty$ with $\alpha = 2$ in step-limited PPR has a negligible effect on the downstream task, while increasing the bitrate by a factor of ≈ 2; see the bottom left panel of Figure 1. In Appendix F, we show that this factor is explained by comparing the KL divergence between the original Gaussians and their moment-matched Laplace counterparts.

5 Experiments

We compare DP-DiPP with two privatize-then-compress baselines on CIFAR-10 and on the Shenzhen chest X-ray set (Jaeger et al., 2014), measuring utility as the test performance of a classifier trained on the privatized images. The *Laplace* baseline adds independent Laplace noise per channel with

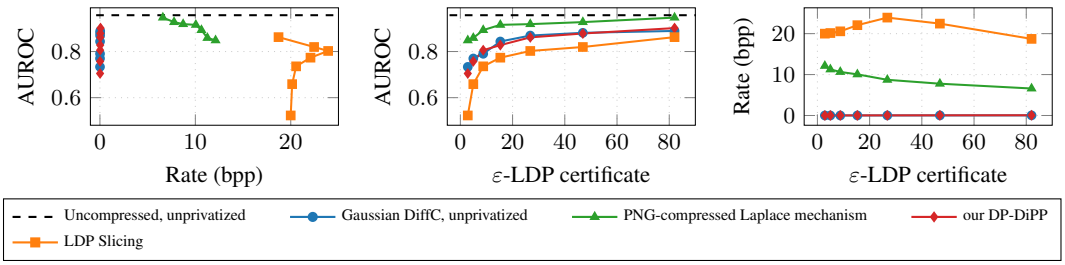

Figure 2: Comparison of experimental results for training a binary classifier from a dataset of privatized and compressed Shenzhen Chest X-ray images. Markers are connected by lines **in order of increasing** epsilon, where ϵ is the method's LDP guarantee, except for DiffC results, which are only included for performance comparison. **Left column:** The rate-utility tradeoff, as measured by bits per pixel (bpp) and the classifier's AUROC, respectively. **Middle column:** Privacy-utility tradeoff, where the method's LDP certificate measures privacy. **Right column:** Rate-privacy tradeoff.

the per-pixel budget split equally over the C channels, denoises the result with the diffusion model from the time-step whose noise variance matches, and stores the released image as a lossless `PNG`; the reported rate is the size of the released `PNG`. The *LDP-Slicing* baseline (Cao et al., 2026) uses the authors' code with a one-level Haar transform and the 4:1:1 budget allocation, with LL pruning. Greyscale images are handled as described in Appendix G.

5.1 Privatization

For privatization, our encoding pipeline modifies DiffC in two ways: we sample from the moment-matched Laplace channel ($b = \sigma/\sqrt{2}$), and we compress this channel with PPR using $\alpha = 2$. At each encoding step, our CUDA kernel samples from $z_i \overset{\text{iid}}{\sim} \mathcal{L}(0, 1)$ and computes the logarithm of density ratios evaluated at those samples as $\sum_i (|z_i| - |z_i - \mu_i|)$ (except the constant term), differently from the implementation in Vonderfecht & Liu (2025) for PFR, which is $\sum_i z_i \mu_i$ (here $\mu_i = (\mu_q - \mu_p)/b$). The encoder then uses these precomputed density ratios to run step-limited PPR (Definition 2). To decode, the receiver, sharing the same random seed, retrieves the selected sample and reconstructs $x_s = b \cdot z + \mu_p$ using the diffusion model's noise prediction

$$\mu_p(x_t, \eta_\theta; s, t) = \frac{1}{\gamma_{t|s}} \left(x_t - \frac{\sigma_{t|s}^2}{\sigma_t} \eta_\theta(x_t, t) \right)$$

Each encoded image produces two outputs. The *noisy reconstruction* is the raw output from the final encoding step; it lies at an intermediate diffusion time-step and contains visible noise. The *denoised reconstruction* is obtained by running DDPM from this final time-step to $t = 0$. For the Laplace noising baseline, we add independent $\mathcal{L}(0, \Delta'_{\text{ch}}/(\epsilon/3))$ noise to each channel where $\Delta'_{\text{ch}} = 255$, clamp to $[0, 255]$, and compress losslessly with `PNG`. Encoding schedule selection for different methods is detailed in Appendix I.

5.2 Classification

The pretrained DDPM backbone (Ho et al., 2020) was trained on the CIFAR-10 training set. To prevent data leakage, we use the CIFAR-10 test set, split evenly into 5k training and 5k evaluation images. To measure how privacy affects downstream utility, we follow Cao et al. (2026) and train a ResNet-56 (He et al., 2016) classifier directly on the privatized images. The classifier never sees clean data; we train it in the noisy domain using the provided image labels and the same hyperparameters as Cao et al. (2026) to enable direct comparison.

For chest X-ray experiments, we train a DenseNet-121 (Huang et al., 2017) model initialized from an ImageNet-trained checkpoint. We evaluate both models on privatized test images and report ablations on the clean image dataset in Appendix H.

5.3 THE DIFFC BASELINE

We compare DP-DiPP to the original Gaussian DiffC (Vonderfecht & Liu, 2025), which is equivalent to our step-limited PPR algorithm with $\alpha = \infty$. Note that PPR with $\alpha = \infty$ lacks a pure LDP guarantee; we include it as an ablation to show that DP-DiPP's performance does not degrade significantly when using moment-matched Laplace distributions instead of the original Gaussians. In CIFAR-10 experiments, to ensure a fair comparison between DiffC and DP-DiPP, we first solve the resource-constrained shortest path problem for the DP-DiPP mechanism at each privacy level ϵ. After we decide on a final time-step T_{final}, we find the shortest path for DiffC that leads to T_{final}. For chest X-ray experiments, we keep the time-step schedule selection the same for both methods.

5.4 EMPIRICAL FINDINGS

Our experimental results on CIFAR-10 are shown in Figure 1. Each point on the curve represents a final compression time-step, after which each method uses the DDPM denoiser to complete denoising the images. For our experiments, we tuned each method to attain a per-pixel ϵ-LDP certificate (or equivalent in the case of DiffC) for $\epsilon \in \{1, 4, 16, 64, 256\}$. For the step-limited PPR code in DP-DiPP, we set $\alpha = 2$.

First and foremost, our findings validate our method: DP-DiPP's joint privatization and compression significantly outperforms the privatize-then-compress baseline, as shown in the top row of Figure 1. In particular, DP-DiPP achieves the same classification accuracy (top left panel) and the same LDP guarantee (top right panel) using less than 3-11% of the bits that the baseline method needs, yielding a 10-30$\times$ improvement in the rate! Second, in the bottom row of Figure 1, comparison of DiffC and DP-DiPP shows that using moment-matched Laplace distributions instead of Gaussians and a strict privacy parameter of $\alpha = 2$ introduces a roughly 2-times penalty in the rate (bottom right), while the utility is largely unaffected.

Figure 2 shows the chest X-ray results, for per-pixel budgets form 2.8 to 82. DP-DiPP releases cost 0.013 to 0.059 bpp against the 6.6 to 12.1 bpp for the Laplace baseline and 18.7 to 23.9 bpp for LDP-Slicing, a 110-920$\times$ and 310-1500$\times$ reduction, respectively, at the same certificate, while the classifier reaches an AUROC of 0.70 at the smallest budget and 0.90 at the largest, against 0.85-0.95 for the Laplace baseline and 0.52-0.86 for LDP-Slicing. DiffC on the same schedule costs half the bits, with an AUROC within one standard deviation of DP-DiPP's at every budget, confirming the factor-two analysis of Appendix F.

Note for both the CIFAR and chest X-ray image comparisons, the PNG-compressed Laplace mechanism appears to yield roughly the same accuracy at a $\times 4$ lower ϵ LDP guarantee. Recall, that this is due to the fact that step-limited PPR for $\alpha = 2$ yields a $\times 4$ looser LDP guarantee than the guarantee of the original mechanism (Theorem 3.1 part 1). However, the obvious empirical attack on the privatized code is to decode the output of the privacy mechanism it encodes and attempt to reconstruct the data from the decoded output. Under this attack, the empirical attack guarantee of DP-DiPP thus perform essentially the same as that of the original Laplace mechanism.

6 CONCLUDING REMARKS AND FUTURE DIRECTIONS

We introduced DP-DiPP, which privatizes and compresses images jointly. It successively transmits Laplace-modified reverse process of a diffusion model using step-limited PPR, which keeps the PPR's privacy guarantee and code rate while significantly reducing the runtime. Using Laplace instead of Gaussian denoising distributions allows DP-DiPP to provide formal $\epsilon-$LDP guarantee while retaining empirical performance. At matched pixel-level ϵ, DP-DiPP reduces the compression rate drastically relative to privatize-then-compress baselines, and it outperforms LDP-Slicing in utility. We note that we have considered pixel-level guarantees, which do not necessarily prevent recognition of global content. Also, we have only dealt with pure ϵ-LDP mechanisms. Thus, a future theoretical direction could be to consider whether it is possible to construct approximate versions of PPR from (ϵ, δ)-LDP mechanisms.

DISCLOSURE OF USING AI

We used generative AI tools to assist with drafting and revising the manuscript, checking numerical comparisons against experimental outputs, and discussing the interpretation and limitations of the results. The authors reviewed the AI-assisted material and take full responsibility for the accuracy and integrity of the final manuscript, including its mathematical claims, experimental results, and references.

REFERENCES

Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 901–914, 2013.

Borja Balle and Yu-Xiang Wang. Improving the Gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International conference on machine learning*, pp. 394–403. PMLR, 2018.

Yuanming Cao, Chengqi Li, and Wenbo He. LDP-Slicing: Local differential privacy for images via randomized bit-plane slicing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Wei-Ning Chen, Peter Kairouz, and Ayfer Ozgur. Breaking the communication-privacy-accuracy trilemma. *Advances in Neural Information Processing Systems*, 33:3312–3324, 2020.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 429–438, 2013. doi: 10.1109/FOCS.2013.53.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and trends® in theoretical computer science*, 9(3-4):211–487, 2014.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284. Springer, 2006.

Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 1054–1067, 2014.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Vitaly Feldman and Kunal Talwar. Lossless compression of efficient private local randomizers. In *International Conference on Machine Learning*, pp. 3208–3219. PMLR, 2021.

Gergely Flamich. *Data Compression with Relative Entropy Coding*. PhD thesis, Apollo - University of Cambridge Repository, 2024. URL `https://www.repository.cam.ac.uk/handle/1810/385303`. PhD Thesis.

Gergely Flamich and Deniz Gündüz. Data compression with stochastic codes. *IEEE BITS the Information Theory Magazine*, pp. 1–14, 2026. doi: 10.1109/MBITS.2026.3702602.

Gergely Flamich and Lennie Wells. Some notes on the sample complexity of approximate channel simulation. In *2024 IEEE International Symposium on Information Theory Workshops (ISIT-W)*, pp. 1–6. IEEE, 2024.

Burak Hasırcıoğlu and Deniz Gündüz. Communication efficient private federated learning using dithering. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7575–7579, 2024. doi: 10.1109/ICASSP48485.2024.10446222.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 2261–2269. Ieee, 2017.

Stefan Jaeger, Sema Candemir, Sameer Antani, Yi-Xiáng J. Wáng, Pu-Xuan Lu, and George Thoma. Two public chest X-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative Imaging in Medicine and Surgery*, 4(6):475–477, December 2014. doi: 10.3978/j.issn.2223-4292.2014.11.20.

Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1376–1385, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/kairouz15.html.

Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint arXiv:2510.21890*, 2025.

Cheuk Ting Li and Abbas El Gamal. Strong functional representation lemma and applications to coding theorems. *IEEE Transactions on Information Theory*, 64(11):6967–6978, 2018.

Cheuk Ting Li et al. Channel simulation: Theory and applications to lossy compression and differential privacy. *Found. and Trends in Comm. and Info. Theory*, 21(6):847–1106, 2024.

Michael F Liu, Saiyue Lyu, Margarita Vinaroz, and Mijung Park. Differentially private latent diffusion models. *arXiv preprint arXiv:2305.15759*, 2023.

Yanxiao Liu, Wei-Ning Chen, Ayfer Ozgur, and Cheuk Ting Li. Universal exact compression of differentially private mechanisms. *Advances in Neural Information Processing Systems*, 37: 91492–91531, 2024.

Abhin Shah, Wei-Ning Chen, Johannes Balle, Peter Kairouz, and Lucas Theis. Optimal compression of locally differentially private mechanisms. In *International Conference on Artificial Intelligence and Statistics*, pp. 7680–7723. PMLR, 2022.

Ali Moradi Shahmiri, Chih Wei Ling, and Cheuk Ting Li. Communication-efficient laplace mechanism for differential privacy via random quantization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4550–4554, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Sreejith Sreekumar and Deniz Gündüz. Optimal privacy-utility trade-off under a rate constraint. In *IEEE International Symposium on Information Theory (ISIT)*, pp. 2159–2163, 2019. doi: 10.1109/ISIT.2019.8849330.

Lucas Theis, Tim Salimans, Matthew D Hoffman, and Fabian Mentzer. Lossy compression with Gaussian diffusion. *arXiv preprint arXiv:2206.08889*, 2022.

Aleksei Triastcyn, Matthias Reisser, and Christos Louizos. DP-REC: Private & communication-efficient federated learning. *arXiv preprint arXiv:2111.05454*, 2021.

United States Census Bureau. Disclosure avoidance for the 2020 census: An introduction. *Handbook, US Government Publishing Office, Washington, DC*, 2021.

Jeremy Vonderfecht and Feng Liu. Lossy compression with pretrained diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 687–702, 2025.

A NOTATION

We denote the binary logarithm as $\log$. For a finite set A, we denote the number of its elements as $|A|$, for a string $s \in \{0,1\}^*$, we denote its length by $|s|$. $\mathbf{1}[\cdot]$ denotes the indicator function. For a real-valued function f, we denote its supremum norm as $\|f\|_\infty = \sup_x |f(x)|$. For two random variables X and Y and a probability distribution P, we write $X \sim P$ to mean that X has distribution P, and write $X \sim Y$ to mean that the variables are equal in distribution. We denote uniform distributions over the interval (a, b) as $\mathrm{Unif}(a, b)$, exponential distributions with rate λ as $\mathrm{Exp}(\lambda)$ and normal distributions with mean μ and variance σ^2 as $\mathcal{N}(\mu, \sigma^2)$. Furthermore, we write $X \perp Y$ to denote the independence of X and Y. For two probability distributions Q and P, we denote the relative entropy/Kullback-Leibler divergence of Q from P as $D_{\mathrm{KL}}[Q \parallel P]$ (in bits) and their total variation distance as $D_{\mathrm{TV}}[Q \parallel P]$. For random variables X, Y, we denote their mutual information as $I[X\ ;\ Y]$ (in bits). Finally, $\mathcal{O}(\cdot)$ is the standard big O notation.

B DESCRIPTION OF THE DIFFC BACKBONE IMPLEMENTATION

Beyond porting DiffC to large pretrained models, Vonderfecht & Liu (2025) introduced several engineering contributions that reduce encoding times for high-resolution images to below 10 seconds. They perform relative entropy coding in fixed-size chunks of approximately 16 bits of KL, splitting high-KL transitions across statistically independent subsets of dimensions, so that each chunk can be feasibly compressed with a stochastic code. Furthermore, they use a custom CUDA kernel for the stochastic code, which draws the candidate samples directly inside the kernel rather than materializing them in GPU memory. Finally, they cast the choice of denoising time-step schedule as a shortest path problem over the cost matrix of pairwise per-step KL divergences. As an ablation, they show that hard-coding the per-encoding-step KL budget across all images, rather than transmitting it as side information, has a negligible effect on the resulting rate-distortion curves.

C PROOF OF THEOREM 3.1

Proof. **LDP guarantee.** Let x, x' be two inputs. Since the mechanism M is ϵ-LDP by assumption, we have

$$e^{-\epsilon} r(m \mid x') \le r(m \mid x) \le e^{\epsilon} r(m \mid x') \tag{15}$$

Next, by a standard fact about the argmin of scaled exponential random variables (see Section 3.3 of Li et al. (2024), for example), we have

$$\begin{aligned}
\mathbb{P}[K_n^\alpha(x, Z, V) = k \mid Z] &= \frac{T_k^\alpha \cdot r(M_k \mid x)^{-\alpha}}{\sum_{j=1}^n T_j^\alpha \cdot r(M_j \mid x)^{-\alpha}} \\
&\le \frac{T_k^\alpha \cdot (e^{-\epsilon} r(M_k \mid x'))^{-\alpha}}{\sum_{j=1}^n T_j^\alpha \cdot (e^{\epsilon} r(M_j \mid x'))^{-\alpha}} \\
&= e^{2\alpha\epsilon} \frac{T_k^\alpha \cdot r(M_k \mid x')^{-\alpha}}{\sum_{j=1}^n T_j^\alpha \cdot r(M_j \mid x')^{-\alpha}} \\
&= e^{2\alpha\epsilon} \mathbb{P}[K_n^\alpha(x', Z, V) = k \mid Z]
\end{aligned}$$

as desired.

Codelength. By its definition $K_n^\alpha(x, z, v) \le K(x, z, v)$, and hence we have that

$$\mathbb{E}[\log K_n^\alpha(x, Z, V)] \le \ell_\alpha(x) \tag{16}$$

Thus, using our Shannon-Fano code ζ to encode $K_n^\alpha(x, Z, V)$, we get the desired result.

Approximation quality. We use a technique similar to that of Flamich & Wells (2024): Let $K(X, Z, V)$ be the index of exact PPR defined in Equation (3). First, note that (e.g., using the technique from Section 4.3.2 of Flamich (2024)):

$$D_{\mathrm{TV}}[Q^{(n)}_{M|X=x} \parallel P_{M|X=x}] \le \mathbb{P}[K(X, Z, V) > k]$$

By Theorem 4.3 of Liu et al. (2024) we have for $\ell(X) = D_{\mathrm{KL}}[P_{M|X} \parallel Q_M]$ that

$$\mathbb{E}[\log K_n^\alpha(X, Z, V)] \leq \ell_\alpha(X) \tag{17}$$

Since $\log K(X, Z, V) \geq 0$, we may apply Markov's inequality to obtain

$$\mathbb{P}[K(X, Z, V) > n] \leq \frac{\ell_\alpha(X)}{\log(n+1)} \tag{18}$$

Therefore, we see that

$$D_{\mathrm{TV}}[Q_{M|X=x}^{(n)} \parallel P_{M|X=x}] \leq \frac{\ell_\alpha(X)}{\log(n+1)} \tag{19}$$

Thus, setting $n \geq 2^{\frac{\ell_\alpha(X)}{\beta}}$ yields the desired approximation guarantee. □

D Notes on Privacy Guarantee of Gaussian Diffusion

We can investigate the privacy guarantee for the Gaussian diffusion process, continuing from Equation (12). The ℓ_2 sensitivity of this mechanism is

$$\begin{aligned}\Delta_{t\to s} &= \frac{\gamma_s \sigma_{t|s}^2}{\sigma_t^2} \sup_{x_0, x_0' \in \mathcal{X}} \|x_0 - x_0'\|_2 \\ &= \frac{\gamma_s \sigma_{t|s}^2}{\sigma_t^2} D_{\mathcal{X}}.\end{aligned}$$

and the Δ/σ ratio simplifies to

$$\begin{aligned}\frac{\Delta_{t\to s}}{\sigma(s,t)} &= \frac{\gamma_s \sigma_{t|s}^2/\sigma_t^2}{\sigma_{t|s}\sigma_s/\sigma_t} D_{\mathcal{X}} \\ &= \frac{\gamma_s \sigma_{t|s}}{\sigma_t \sigma_s} D_{\mathcal{X}} \\ &= \sqrt{\left(\frac{\gamma_s}{\sigma_s}\right)^2 - \left(\frac{\gamma_t}{\sigma_t}\right)^2} D_{\mathcal{X}} \\ &= \sqrt{\mathrm{SNR}(s) - \mathrm{SNR}(t)} D_{\mathcal{X}}.\end{aligned}$$

The (ϵ, δ)-differential privacy of this mechanism at each time step can be computed tightly using the result of Balle & Wang (2018, Theorem 8). The privacy guarantee of the full multi-step mechanism, which is the composition of per-step mechanisms, can then be obtained via an adaptive composition theorem such as Kairouz et al. (2015, Theorem 3.5).

E Custom CUDA Kernel for PPR

Different from the implementation released by the authors of Vonderfecht & Liu (2025), to sample from a moment matched laplace distribution at each time-step, we first generate laplace samples with mean 0 and scale 1. We draw these samples using inverse transform sampling:

```
__device__ float curand_laplace_standard(curandState* state) {
    float u = curand_uniform(state);
    float half_minus = u - 0.5f;
    float sign_val = (half_minus >= 0.0f) ? 1.0f : -1.0f;
    float abs_val = fabsf(half_minus);
    float inner = fmaxf(1.0f - 2.0f * abs_val, 1e-10f);
    return -sign_val * logf(inner);  // Laplace(0, 1)
}
```

Also differently from Vonderfecht & Liu (2025), the log density ratio ($\log w(z)$) between target $\mathcal{L}(\mu, 1)$ (where $\mu = \frac{\mu_q - \mu_p}{b}$) and proposal $\mathcal{L}(0, 1)$ analytically becomes

$$\log w(z) = |z| - |z - \mu|$$

thus we generate these ratios as:

```
    float log_w_value = 0.0f;
    for (int i = 0; i < dim; i++) {
        float z = curand_laplace_standard(&state);
        log_w_value += fabsf(z) - fabsf(z - mu_q[i]);
    }
```

We decide on the sample returned by step limited PPR using:

```
    t = cp.random.exponential(scale=1.0, size=K)
    v = cp.random.exponential(scale=1.0, size=K)

    s = alpha * (log_cumsum_t - log_w) + log_v

    winning_seed = cp.argmin(s).item()
    sample = generate_sample(dim, shared_seed, winning_seed)
```

We note that we can convert samples from $Z_i \overset{\text{iid}}{\sim} \mathcal{L}(\frac{\mu_q - \mu_p}{b}, 1)$ to $X_i \overset{\text{iid}}{\sim} \mathcal{L}(\mu_q, b)$ by

$$X_i = Z_i \cdot b + \mu_p,$$

where both μ_p and b are available to both the encoder and decoder. This is implemented in the code as

```
            mu = ((qmu[i] - pmu[i]) / b).flatten().detach().cpu().numpy()
            sample, cs, dkl = ch_sim.encode(mu, manual_dkl=manual_dkl[si],
            ↪  seed=si)
            seeds_all[i].append(cs); dkl_all[i].append(dkl)
            xl[i] =
            ↪  torch.tensor(sample).reshape(xl[i].shape).to(dev).to(dt)
            ↪  * b + pmu[i]
```

F Explanation of the Factor Two Increase in Bitrate after Moment Matching

We illustrate the phenomenon for one-dimensional distributions. As the distributions involved in practice are fully factorized dimensionwise, the conclusion carries over by the chain rule of relative entropies.

Thus, consider now

$$q = \mathcal{N}(\mu_q, \sigma^2), \quad p = \mathcal{N}(\mu_p, \sigma^2),$$
$$q' = \mathcal{L}(\mu_q, \sigma/\sqrt{2}), \quad p' = \mathcal{L}(\mu_p, \sigma/\sqrt{2})$$

where q, p represent one dimension of the conditional and unconditional denoising distributions from Equation (6), and q', p' are their moment-matched counterparts. For convenience, let $\delta = \frac{|\mu_p - \mu_q|}{\sigma}$ be the standardized absolute mean deviation. Then, using the well-known formulae for the relative entropy of two Gaussian and two Laplace distributions, we get

$$D_{\mathrm{KL}}[q \parallel p] \cdot \ln 2 = \frac{\delta^2}{2} \tag{20}$$

$$D_{\mathrm{KL}}[q' \parallel p'] \cdot \ln 2 = \sqrt{2}\delta - 1 + e^{-\sqrt{2}\delta} \tag{21}$$

$$= \sqrt{2}\delta - 1 + 1 - \sqrt{2}\delta + \delta^2 + \mathcal{O}(\delta^3) \qquad \text{(Taylor expansion around 0)}$$

$$= \delta^2 + \mathcal{O}(\delta^3) \tag{22}$$

Comparing Equations (20) and (22), we see that for small δ, we indeed have $D_{\mathrm{KL}}[q' \parallel p'] \approx 2 \cdot D_{\mathrm{KL}}[q \parallel p]$, as we observe in practice.

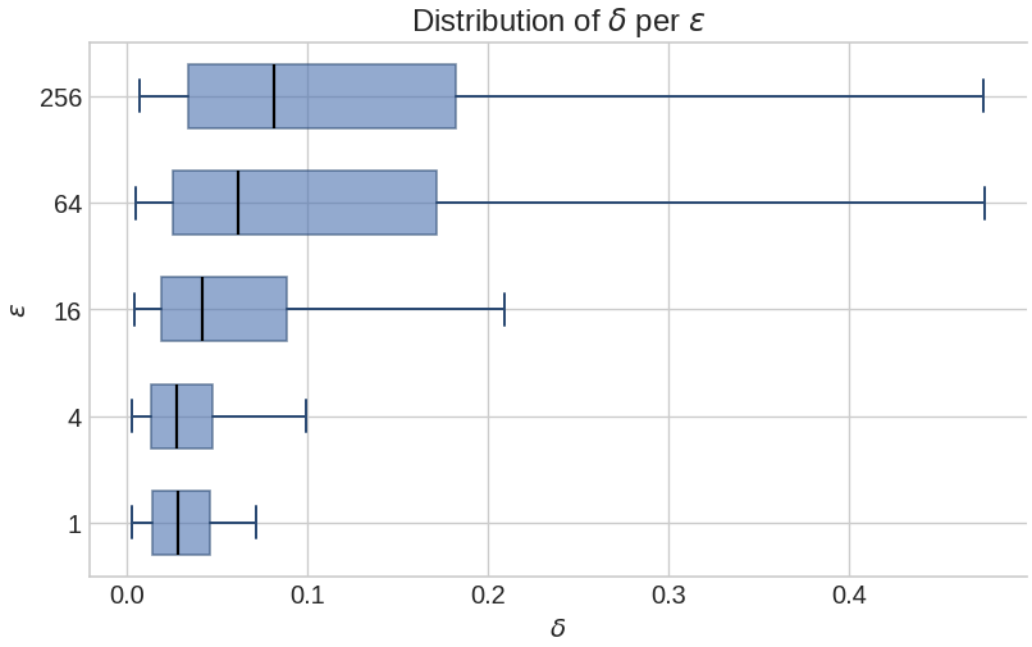

Figure 3: Per ϵ distribution of the standardized absolute mean deviation δ. Each box pools δ over all time steps of the PPR schedule, all 128 test images, and all $32 \times 32 \times 3$ latent dimensions. Box = interquartile range, line = median, whiskers = 5th/95th percentiles.

We corroborate this empirically, by studying the distribution of δ across the dimensions of our diffusion models for different ϵ-LDP settings in Figure 3. We see that for all settings, the distribution of δ is heavily skewed towards 0, with barely any values being close even to 1; thus our Taylor approximation is roughly valid.

G GREYSCALE IMAGES

For the greyscale X-rays, we copy each pixel's intensity to all three color channels and apply each privatization method exactly as for color images. We do this for two reasons. First, LDP-Slicing does not support greyscale images, so a common three-channel representation lets us compare all methods on the same downstream task. Second, our downstream classifiers are initialized from models pretrained on color images and therefore expect three-channel inputs. Accordingly, for the baselines, we report lossless compression rates for the resulting three-channel images.

H ERROR BARS AND TEST SET ABLATIONS

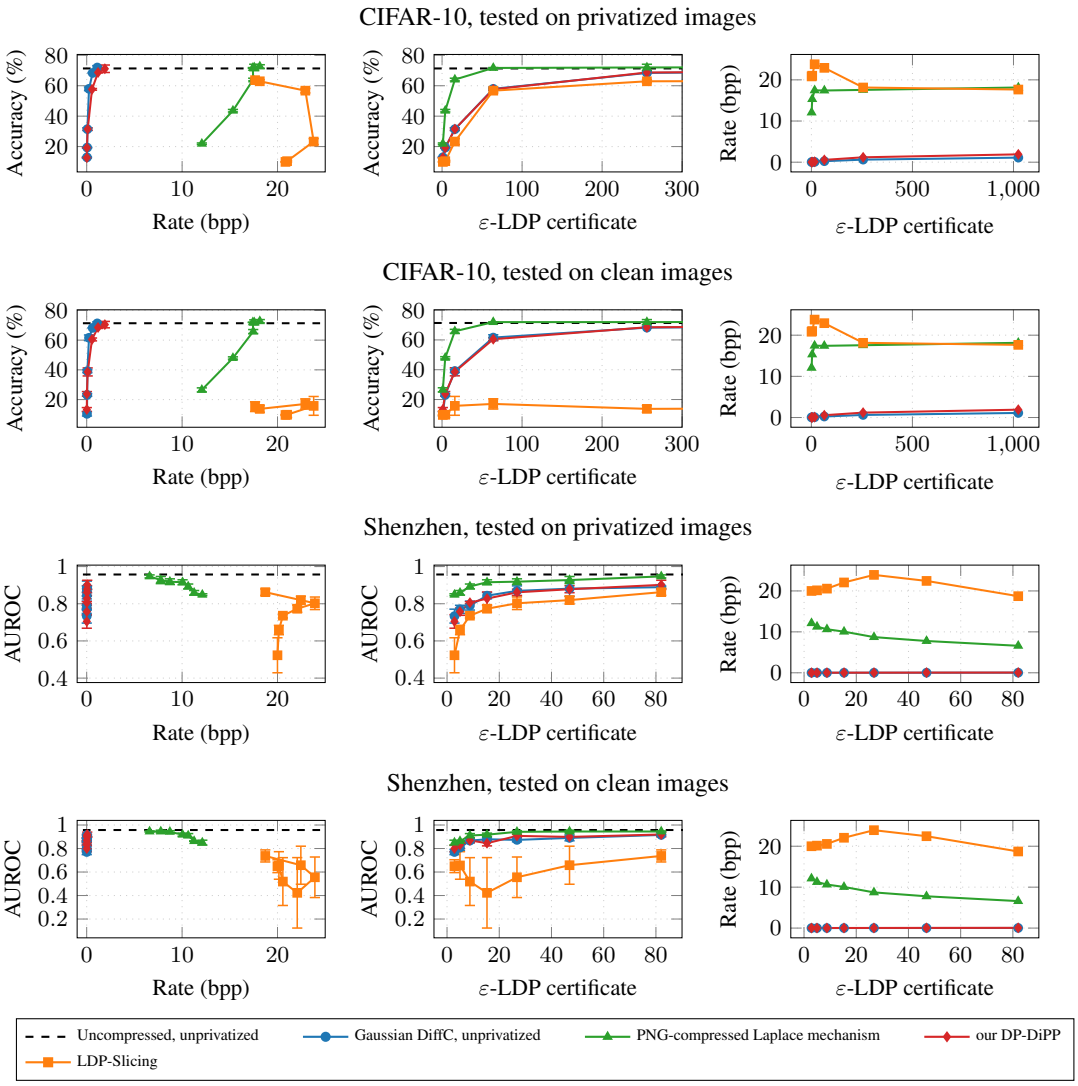

Figure 4: Experimental results on training an image classifier from a dataset of privatized and compressed images. **First row:** CIFAR-10, ResNet-56 accuracy on privatized test images. **Second row:** CIFAR-10, the same classifiers evaluated on the original, unprivatized images. **Third row:** Shenzhen chest X-rays, DenseNet-121 AUROC on privatized test images. **Fourth row:** Shenzhen chest X-rays, the same classifiers evaluated on the original, unprivatized test images. Note that the Gaussian DiffC **does not** have an LDP certificate and is included only for illustration. The dashed line is the utility of a classifier trained and tested on the original, uncompressed, and unprivatized images. In each case, the markers are connected by lines **in order of increasing** ϵ, where ϵ is the method's per-pixel LDP guarantee. Error bars show the standard deviation over three runs (CIFAR-10: three training seeds; Shenzhen: three random train/validation/test splits). **Left column:** The rate-utility tradeoff, as measured by bits per pixel (bpp) and test utility, respectively. **Middle column:** Privacy-utility tradeoff, where the method's LDP certificate measures privacy. **Right column:** Rate-privacy tradeoff; the rate is the mean over all encoded images and does not depend on the test set. Each panel compares 1) the privatize-then-compress Laplace mechanism, 2) Gaussian DiffC, 3) LDP-Slicing, and 4) DP-DiPP.

I ENCODING SCHEDULE SELECTION

For CIFAR-10 experiments, we choose the encoding schedule by solving a resource-constrained shortest-path problem over the time-steps, extending the method in Vonderfecht & Liu (2025), which does not include a privacy constraint. The aim is to find the schedule that minimizes the total coding cost subject to the privacy budget ϵ_{total} for each final time-step T_{final}.

To come up with the time-step schedule, we find the smallest T_n satisfying

$$\Delta\epsilon_n \coloneqq \epsilon_n - \epsilon'_{n-1} \geq 2\alpha\Delta\sqrt{2(\text{SNR}(T_n) - \text{SNR}(T_{n-1}))}$$

and iteratively refine our $\epsilon_n \to \epsilon'_n$ so that $\epsilon'_n - \epsilon'_{n-1} = 2\alpha\Delta\sqrt{\text{SNR}(T_n) - \text{SNR}(T_{n-1})}$. As the initial condition, we have $\epsilon_0 = 0$ and $T_0 = 1000$. We did not observe a considerable gain in performance in terms of PSNR when $\bar{\alpha}_{T_0}$ is fixed as $\bar{\alpha}_{T_0} = 0$ as compared to it being determined by the noise schedule of the model. Since the model-dictated schedule yields better privacy guarantees, we chose this option.

For chest X-ray experiments, to generate privatized samples that have coverage over different utility regimes, we select the time-steps such that the cumulative privacy leakage until that time-step is approximately given by

$$\epsilon_n = a_0^{n-1}\epsilon_{\text{min}}$$

where $n \in \{1, \cdots, N\}$ and N is the number of privacy values with which we evaluate our model. We choose ϵ_{min} and a_0 empirically based on the visual coverage.

J DETAILS FOR THE EXPERIMENTS

J.1 DATASETS, MODELS, AND DOWNSTREAM TASKS

J.1.1 CIFAR-10

We use the CIFAR-10 test split of 10,000 images with resolution $32 \times 32 \times 3$ as our dataset, using the `google/ddpm-cifar10-32` Hugging Face model and checkpoints trained on the training split as the pretrained diffusion backbone for DP-DiPP. We divide the dataset into class-stratified halves of 5,000 training and 5,000 test images, and hold out the last 10% of each class in the training half for validation.

For DP-DiPP and DiffC, we estimate the per-step KL budgets in advance on a set of CIFAR-10 training images and hard-code them for use during encoding (Section J.2). We report compression rate as the encoded bitstream length divided by 32×32, averaged over encoded images.

For the downstream task, we train a ResNet-56 from scratch on the privatized training images, following Cao et al. (2026). We use SGD with momentum 0.9, initial learning rate 0.1 halved at epochs 80, 160, and 200, weight decay 10^{-4}, gradient clipping at norm 0.5, batch size 128, 250 training epochs, and random-crop and horizontal-flip augmentation. We keep the checkpoint with the best validation accuracy and evaluate it on both the privatized and clean test images (Figure 4). We repeat each experiment with three training seeds and report the mean and standard deviation of the test accuracy.

J.1.2 SHENZHEN

We use the Shenzhen Hospital chest X-ray dataset (Jaeger et al., 2014) of 662 images, with 336 tuberculosis and 326 normal diagnosis labels. We convert each image to greyscale, zero-pad it to a square, resize it to 512×512 using antialiased bilinear interpolation, and copy the greyscale image to three channels. We treat the resulting images as colour images for the rest of the experiments (Appendix G).

We create three random splits of the dataset into 369 training, 93 validation, and 200 test images, stratified by diagnosis, sex, and age band from the clinical readings. For DP-DiPP and DiffC, we use the pixel-space ADM 512×512 model of Dhariwal & Nichol (2021) (`openai/guided-diffusion`) as the pretrained diffusion backbone. The model is class-conditional and is run with a fixed class label. Since pixels are scaled to $[-1, 1]$, the per-channel sensitivity is 2.

Both codecs use the encoding timestep schedule $1000 \to 920 \to 893 \to 834 \to 740 \to 610 \to 451 \to 286$ (Appendix I), where the release at the n-th timestep corresponds to the n-th privacy level. We calibrate the per-step KL budgets on the disjoint Montgomery chest X-ray dataset of 138 images, separately for the Gaussian and Laplace channels. We report compression rate as the encoded bitstream length divided by 512×512, averaged over all 662 images.

For the downstream task, we use binary classification and fine-tune an ImageNet-pretrained DenseNet-121 on the three-channel released training images, resized to 224×224. We train with AdamW using a learning rate and weight decay of 10^{-4}, a cosine schedule, a batch size of 32, small random affine augmentation, and at most 100 epochs with early stopping after 15 epochs without improvement. We keep the checkpoint with the best validation AUROC and choose the decision threshold that maximizes validation balanced accuracy. We then evaluate each classifier on the 200 test images privatized with the same method and on the clean test images. We report the mean and standard deviation of the test AUROC over the three splits.

J.2 D_{KL} PER STEP CALCULATIONS

After fixing the timestep schedule, we empirically estimate the per-step KL divergence between the target distribution $q(x_s \mid x_t, x_0)$ and the diffusion model proposal $p_\theta(x_s \mid x_t)$ using a calibration set that is disjoint from the encoded images. We run the encoder along the full schedule for each calibration image, compute the KL divergence at every step using the closed-form expression for the Gaussian or Laplace channel, and hard-code the average KL at each step over the calibration images.

The encoder then uses these fixed KL budgets for every image. Fixing the KL budgets in advance is important for a realistic DiffC compression setting and is essential for DP-DiPP, since computing the budget from the encoded image would make it depend on x_0 and introduce additional privacy leakage beyond the certified ϵ.