Synthesizing Differentially Private Datasets using Random Mixing

Kangwook Lee, Hoon Kim, Kyungmin Lee, and Changho Suh
School of EE, KAIST
Email: {kw1jjang, gnsrla12, atm13579, chsuh}@kaist.ac.kr

Kannan Ramchandran
Dept. of EECS, UC Berkeley
Email: kannanr@eecs.berkeley.edu

***Abstract*—The goal of differentially private data publishing is to release a modified dataset so that its privacy can be ensured while allowing for efficient learning. We propose a new data publishing algorithm in which a released dataset is formed by mixing ℓ randomly chosen data points and then perturbing them with an additive noise. Our privacy analysis shows that as ℓ increases, noise with smaller variance is sufficient to achieve a target privacy level. In order to quantify the usefulness of our algorithm, we adopt the accuracy of a predictive model trained with our synthetic dataset, which we call the utility of the dataset. By characterizing the utility of our dataset as a function of ℓ, we show that one can learn both linear and nonlinear predictive models so that they yield reasonably good prediction accuracies. Particularly, we show that there exists a sweet spot on ℓ that maximizes the prediction accuracy given a required privacy level, or vice versa. We also demonstrate that given a target privacy level, our datasets can achieve higher utility than other datasets generated with the existing data publishing algorithms.**

I. Introduction

In a wide variety of machine learning applications, the training dataset consists of sensitive data such as medical records, personal photos or proprietary data. It is known that a model trained with standard machine learning algorithms is subject to data breach [1]. In order to protect data, one may first synthesize a *private* version of the original training dataset and then train a model solely with the private one [2]–[4].

Differential privacy, introduced by Dwork et al. [5], has been adopted as a common notion of privacy by multiple research communities. Based on this notion, various differentially private data publishing algorithms have been proposed in literature, but most of them are inapplicable to high-dimensional data due to their prohibitive computational costs [3]. This calls for computationally efficient algorithms that can be used for high-dimensional data.

In this work, we propose a new data publishing algorithm, which we call *Differentially Private Mix (DPMix)*. DPMix *mixes* more than one data points, perturbs the mixtures with additive noise, and then publish the perturbed mixtures as an output dataset. We call the constant number of data points involved in each mixture the *mixture degree*, denoted by ℓ. DPMix with mixture degree ℓ is denoted by DPMix(ℓ). See Fig. 1 for sample outputs generated from our algorithm.

We characterize the privacy guarantee of DPMix, showing that noise with smaller variance is sufficient to attain a target privacy level as ℓ increases. To show the usefulness of our dataset for the purpose of learning, we characterize how the performance of a model trained with our synthetic dataset, which we call the *utility* of the dataset, behaves as a function of ℓ. We show that one can learn both linear and nonlinear models with datasets generated by DPMix, achieving reasonably good prediction accuracies. See Fig. 2 for visualization. Particularly, for the case of nonlinear models (Fig. 2b), we show that the utility first increases and then decreases as ℓ increases, implying the existence of a sweet spot on ℓ. We show the existence of such a sweet spot via real-data experiments. We also compare the utility of our algorithm with those of the existing data publishing algorithms, which will be detailed soon [6]–[11].

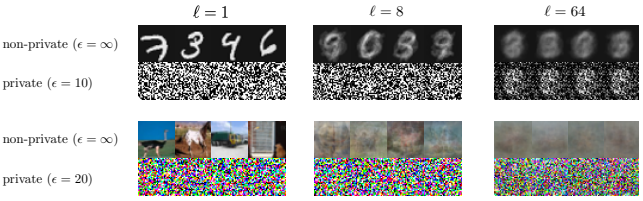
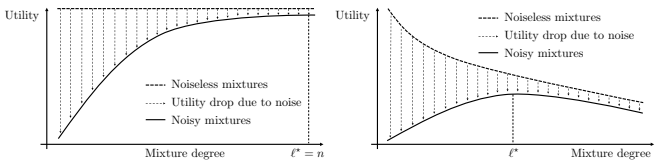


Fig. 1: Differentially private datasets generated out of MNIST and CIFAR10 datasets for a varying mixture degree ℓ. Here, ε denotes the differential privacy parameter; the lower ε is, the higher the privacy level is.

A. Related work

1) Differentially private data publishing algorithms: A variety of differentially private data publishing algorithms have been proposed in literature [12], [13]. For a detailed overview, we refer the readers to a recent survey paper [3]. Unfortunately, the computational complexities of most of the existing algorithms are exponential in the dimensionality of the dataset, so they are not applicable to most deep learning applications that deal with high-dimensional data.

Local perturbation is a simple data publishing algorithm that corrupts every data point with additive noise [6]–[9]. This algorithm is applicable to high-dimensional data due to its low computational complexity. In addition, another advantage of local perturbation is that the synthetic data belongs to the same data domain as the original one. It allows for one to make use of existing learning algorithms that are developed for the original data. For instance, if the original dataset consists of images,



(a) Linear models (b) Nonlinear models

Fig. 2: Utility of our dataset as a function ℓ.

the synthetic dataset also consists of images (although noisy). Thus, with such a synthetic dataset, one may train an efficient deep learning model such as convolutional neural network (CNN). However, local perturbation must inject excessively large additive noise to achieve a high privacy level, making the synthetic dataset barely useful for subsequent learning.

Another data publishing algorithm is based on random projection [10], [11]. This algorithm first extracts lower-dimensional features via random projection and then corrupts them with an additive noise. Note that one cannot make use of domain-specific models or algorithms in this case since the synthetic data lies in a different domain.

2) Differentially private algorithms for machine learning: Differentially private algorithms have been extensively studied for a variety of machine learning problems such as regression [14]–[16], online learning [17], graphical models [18], empirical risk minimization [19]–[22], and deep learning [23]–[25]. This line of work considers a slightly different goal: Instead of designing a differentially private dataset (the *input* end of machine learning pipeline), their goal is to design learning algorithms that can be used to obtain differentially private models (the *output* end of the pipeline). Since any model trained with a differentially private dataset is also differentially private [5], this goal is a relaxed version of that of the differential private data publishing problem.

While differentially private models are relevant to many scenarios, they fail to protect the data under extreme scenarios. For instance, if an adversary has an access to the input end of machine learning systems, e.g., data storages, the training data flowing into the input end would be at stake.

3) Learning with mixtures: Karakus et al. show that a simple linear model can be trained with noiseless mixtures [26]. We will show that one can train a linear model even with *noisy* mixtures using a variant of stochastic gradient descent (SGD).

Learning nonlinear models with mixtures has also been investigated in recent studies [27]–[31]. In [27], Tokozume et al. show that one can train a sound recognition model with mixtures of audio signals. Similarly, a few contemporary studies show that one can train an image classification model with mixtures of images [28]–[30]. While noiseless mixtures of only two or three data points are considered in these studies, we will empirically show that one can train deep neural networks with noisy mixtures of much larger degrees.

B. Notation

Let $[n] := \{1, 2, \dots, n\}$. Let I_n be the identity matrix of size $n \times n$. The i^{th} standard basic vector is denoted by e_i. Adopting the MATLAB notation, $[A_1; A_2] := [A_1^\top \ A_2^\top]^\top$. We write $C \sim \text{Bern}(p)$ if $C \in \{0, 1\}$ is a Bernoulli random variable with parameter p. For the case of a Gaussian random variable with mean μ and variance σ^2, we write $C \sim N(\mu, \sigma^2)$. Similarly, we write $C \sim N(\mu, \Sigma)$ if $C \in \mathbb{R}^d$ is a multivariate Gaussian random variable whose mean vector is $\mu \in \mathbb{R}^d$ and the covariance matrix is $\Sigma \in S_+^d$, where S_+^d is the set of positive semidefinite matrices of size d by d. We use a shorthand notation $\{x_i\}_{i=1}^n$ for $\{x_i : 1 \le i \le n\}$. If not specified otherwise, $\|\cdot\|$ denotes the L_2 norm of a vector or the $L_{2,2}$ norm (or the Frobenius norm) of a matrix.

II. Data publishing algorithm

We introduce our data publishing algorithm: DPMix. Consider the labeled dataset with n data points consisting of a feature matrix $X := [X_1 \ X_2 \ \cdots \ X_n] \in \mathbb{R}^{d_X \times n}$ and a label matrix $Y := [Y_1 \ Y_2 \ \cdots \ Y_n] \in \mathbb{R}^{d_Y \times n}$, where (X_i, Y_i) is the i^{th} data point. The data owner wants to generate a differentially private version of this dataset. For simplicity, we assume that X (Y) is normalized such that $X \in [0, 1]^{d_X \times n}$ ($Y \in [0, 1]^{d_Y \times n}$) and $\|X_i\| \le 1$ ($\|Y_i\| \le 1$) for all $i \in [n]$.

Given the size of synthetic dataset T, DPMix(ℓ) generates T random mixtures of degree ℓ and then corrupts them with an additive noise. That is,

$$(X'_t, Y'_t) = (XC_t + Q_t, YC_t + R_t)\,, \ \forall t \in [T], \qquad (1)$$

where for each t, $C_t = [C_{t,1}; C_{t,2}; \cdots; C_{t,n}]$, and Q_t and R_t are white Gaussian noise processes, i.e., $Q_t \sim N(0, \sigma_X^2 I_{d_X})$ and $R_t \sim N(0, \sigma_Y^2 I_{d_Y})$ for some (σ_X, σ_Y). Here, ℓ entries of C_t are $\frac{1}{\ell}$ while others are 0, and the ℓ entires are chosen uniformly at random, independently of others. The noise parameters (σ_X, σ_Y) are used to control the privacy level of the output dataset. The computational complexity is $\mathcal{O}(T\ell(d_X + d_Y))$. Shown in Fig. 1 are sample mixtures generated from MNIST and CIFAR10 datasets with various values of ℓ and (σ_X, σ_Y).

III. Main Results

In this section, we first show that the output of DPMix is *differentially private*, where the privacy level is determined by ℓ, (σ_X, σ_Y), and T. We also study the usefulness of the private datasets generated by DPMix. Particularly, we are interested in the performance of a predictive model trained with a synthetic dataset, which we call the *utility* of the dataset. For the case of linear models, we prove that the utility of our synthetic dataset increases as ℓ increases, i.e., the optimal mixture degree $\ell^\star$ is n. For the case of nonlinear models, we empirically show that the utility first increases and then decreases as ℓ increases, implying $1 < \ell^\star < n$.

A. Differential privacy guarantees

We first review the concept of *differential privacy* [5]. An algorithm is called differentially private if the distribution of the random output of the algorithm does not alter much when the input data is marginally modified. Clearly, this implies that one cannot guess with high confidence whether or not a

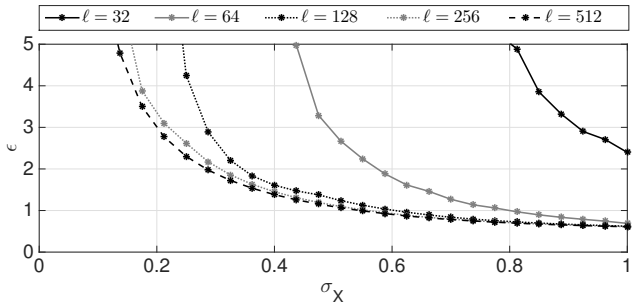


Fig. 3: ε as a function of σ_X for varying values of ℓ. Here, $d_X = d_Y = 50$, $n = T = 10^6$, $\delta = 1/n$, and $\sigma_Y = \sigma_X$.

certain data point is included in the dataset. A formal definition is as follows. Let two datasets $D = \{(X_i, Y_i)\}_{i=1}^n$ and $D' = \{(X_i', Y_i')\}_{i=1}^n$ be adjacent if $|D' \setminus D| = 1$ and $|D \setminus D'| = 1$, i.e., D and D' are identical except for one data point[1]. For adjacent D and D', we write $D \sim D'$. Given this definition, a data publishing algorithm $\mathcal{M}(D)$ is called (ε, δ) differentially private ((ε, δ)-DP for short) if it satisfies

$$\Pr[\mathcal{M}(D) \in A] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in A] + \delta \tag{2}$$

for any $D \sim D'$ and any set of outputs of the mechanism A.

We now present the privacy guarantee of DPMix.

Theorem 1 (Privacy guarantee): Fix the mixture degree ℓ, the noise levels (σ_X, σ_Y) and the number of mixtures T. For any $\delta > 0$, DPMix(ℓ) is (ε, δ)-DP such that

$$\varepsilon = \min_{\alpha \in \{2,3,\ldots\}} T\varepsilon_\alpha' + \frac{\log(1/\delta)}{\alpha - 1}, \tag{3}$$

where

$$\varepsilon_\alpha' = \frac{1}{\alpha - 1} \log \left(1 + \left(\frac{\ell}{n}\right)^2 \binom{\alpha}{2} \min \left(4 \left(e^{\frac{\Delta^2}{\ell^2}} - 1 \right), 2e^{\frac{\Delta^2}{\ell^2}} \right) + 4G(\alpha) \right),$$

$$G(\alpha) := \sum_{j=3}^{\alpha} \left(\frac{\ell}{n}\right)^j \binom{\alpha}{j} \sqrt{B(2\lfloor j/2 \rfloor) \cdot B(2\lceil j/2 \rceil)},$$

$$B(\ell) := \sum_{i=0}^{\ell} (-1)^i \binom{\ell}{i} e^{\frac{i(i-1)}{2\ell^2}\Delta^2}, \quad \Delta^2 := \left(\frac{d_X}{\sigma_X^2} + \frac{d_Y}{\sigma_Y^2} \right).$$

Before we prove the theorem, we first visualize how ε behaves as a function of (σ_X, σ_Y) for varying values of ℓ in Fig. 3. Observe that for a fixed noise level, the privacy guarantee increases with an increase in ℓ. Therefore, if one chooses a larger value of ℓ, then a smaller amount of noise is sufficient to achieve the target privacy level.

Proof: Our proof is a direct consequence of recent developments in Rényi differential privacy [32] and privacy amplification via subsampling [33]. We first give a brief overview of the Rényi divergence and Rényi differential privacy. The Rényi divergence of order $\alpha > 1$ for two distributions P and Q is defined as

$$D_\alpha(P||Q) := \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim Q} \left[\left(\frac{P(x)}{Q(x)} \right)^\alpha \right], \tag{4}$$

[1]Note that some works use a different notion of neighbor datasets: For instance, in [10], [11], neighboring datasets are defined as those that differ only in one entry instead of one data point.

where $P(x)$ and $Q(x)$ are the densities of P and Q evaluated at x, respectively. Rényi differential privacy generalizes the concept of differential privacy based on the Rényi divergence. Particularly, a mechanism $\mathcal{M}(\mathcal{D})$ is said to have ε-Rényi differential privacy of order α ((α, ε)-RDP for short) if $D_\alpha(\mathcal{M}(D)||\mathcal{M}(D')) \leq \varepsilon$ for any two adjacent datasets D and D'. The Rényi differential privacy has some useful properties. First, if a mechanism (α, ε)-RDP, it is $(\varepsilon + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$-DP for any $\delta > 0$ [32, Proposition 3]. That is, any RDP mechanism can be translated into a DP mechanism. Another useful property is about composition: If $\mathcal{M}_1$ is (α, ε_1)-RDP and $\mathcal{M}_2$ is (α, ε_2)-RDP, the simultaneous release $(\mathcal{M}_1, \mathcal{M}_2)$ is $(\alpha, \varepsilon_1 + \varepsilon_2)$-RDP [32, Proposition 1].

We next provide the overview of the proof, deferring the full proof to the appendix. Our proof consists of three steps. First, we show that a single mixture is $(\alpha, \varepsilon_\alpha')$-RDP. To show this, we make use of a recent development in privacy amplification via subsampling [33]. Due to the composition property, T mixtures are $(\alpha, T\varepsilon_\alpha')$-RDP. By converting the RDP guarantee into a DP guarantee, we then have $(T\varepsilon_\alpha' + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$ for any $\delta > 0$. Since this holds for any α, we take the minimum over all possible values of α to yield (3). ■

We remark that when (σ_X, σ_Y) is relatively low, i.e., the low privacy regime, the privacy guarantee in Thm. 1 becomes loose. In this case, a slightly modified version of DPMix can obtain an improved privacy guarantee. More details are provided in the appendix.

B. Learning linear models with a private dataset

Consider a scenario when one tries to learn a linear model, i.e., $Y = WX$. In this section, we will show that our private dataset can be used to train such a linear model. To see why this is the case, consider a degree-2 mixture composed of two data points (X_1, Y_1) and (X_2, Y_2). Assume that the data points perfectly fit with a certain linear model, i.e., $Y_i = WX_i$ for $i \in \{1, 2\}$. Then, the mixture also fits well with the linear model since $Y_1 + Y_2 = W(X_1 + X_2)$. One can easily see that this property holds for any mixture degree. Therefore, one may be able to train this model solely with the private dataset, and the trained model can make predictions in the original domain.

An important question is whether a computationally efficient algorithm like Stochastic Gradient Descent (SGD) or its variants can serve to train a model with a private dataset. The following theorem affirms that a variant of SGD can be used to train a model with a private dataset. We defer the rigorous statements and proofs to the appendix.

Theorem 2 (Convergence guarantee): Consider a cost function $\mathcal{J}(W) = \frac{1}{2n}\|Y - WX\|_F^2$. Under mild conditions, the output of a variant of SGD, denoted by W', satisfies

$$\mathbb{E}[\mathcal{J}(W') - \mathcal{J}(W^\star)] \leq Q(\sigma_X, \sigma_Y)(1 + \log T)/T, \tag{5}$$

where $W^\star = \arg\min_W \mathcal{J}(W)$, $Q(\sigma_X, \sigma_Y)$ is increasing both in σ_X and σ_Y, and the expectation is over randomness of DPMix.

This theorem asserts that for a fixed T, the optimality gap is an increasing function of the noise level. According to Thm. 1,

TABLE I: The test accuracy of CNN on MNIST and CIFAR10.

	MNIST			CIFAR10		
ℓ	$\varepsilon = \infty$	$\varepsilon = 20$	$\varepsilon = 10$	$\varepsilon = \infty$	$\varepsilon = 30$	$\varepsilon = 20$
1	**0.993**	0.098	0.098	**0.713**	0.100	0.100
2	0.982	0.098	0.098	0.637	0.100	0.100
4	0.974	0.098	0.098	0.559	0.100	0.105
8	0.964	0.272	0.098	0.461	0.112	0.117
16	0.949	0.698	0.484	0.374	0.143	0.150
32	0.934	0.765	0.585	0.349	0.156	0.152
64	0.916	**0.800**	**0.782**	0.204	0.204	0.191
128	0.895	0.791	0.764	0.346	**0.269**	**0.244**
256	0.867	0.759	0.691	0.262	0.239	0.222
512	0.839	0.687	0.629	0.188	0.142	0.104

a smaller amount of noise suffices to achieve the same privacy level as ℓ increases. Hence, by choosing $\ell = n$, one can maximize the utility of the dataset, i.e., $\ell^\star = n$.

C. Learning nonlinear models with a private dataset

While it is rather straightforward that one can train linear models with linear mixtures, it is not clear whether one can do so for *nonlinear* models. While several recent studies show that one can train deep neural networks with degree-2 or degree-3 mixtures [27]–[30], it is not clear whether one can train with mixtures of larger degrees.

To answer this, we first generate (*non-private*) synthetic datasets out of MNIST and CIFAR10 datasets with $(\sigma_X, \sigma_Y) = (0, 0)$ [34], [35]. While these datasets are not private, i.e., $\varepsilon = \infty$, the utilities of these datasets can serve as an upper bound on those of private datasets. We also vary the mixture degree by setting $\ell \in \{1, 2, 4, \ldots, 256, 512\}$. We then train a small CNN on each dataset and measure its test accuracy. Note that these performances are measured with respect to the original test dataset, which consists of 'unmixed' data points. More details about the experimental setting are provided in the appendix. The test accuracies are reported in column 'MNIST ($\varepsilon = \infty$)' and 'CIFAR10 ($\varepsilon = \infty$)' of Table I. Observe that one can still train a deep neural network on synthetic datasets with mixtures of very large degree. However, the utility of noiseless mixtures decreases as ℓ increases. For instance, we can achieve the average test accuracy of 0.895 when $\ell = 128$ (See Fig. 1 for sample images) and that of 0.839 when $\ell = 512$ on MNIST.

Let us now consider the utility of private dataset with noisy mixtures. As shown in Thm. 1, the minimum amount of noise required for a fixed privacy level decreases as ℓ increases. That is, the utility reduction due to noise is a decreasing function in ℓ. Therefore, the utility of noisy mixtures might have a sweet spot $\ell^\star$ ($1 < \ell^\star < n$) as shown in Fig. 2b. In order to show the existence of such a sweet spot, we repeat the above experiments with private datasets generated by DPMix. For each value of ℓ, we set (σ_X, σ_Y) such that the synthetic dataset should satisfy the same privacy level. Shown in Table I are the experimental results. One can observe that the best test accuracies are achieved by nontrivial mixture degrees for both datasets. For instance, $\ell^\star = 64$ for the MNIST dataset with $\varepsilon = 20$, and $\ell^\star = 128$ for the CIFAR10 dataset with $\varepsilon = 30$.

TABLE II: Performance Comparisons. The first four two are for DPMix($\ell^\star$) and the other four rows are for baselines.

	MNIST		CIFAR10	
Algorithm	$\varepsilon = 20$	$\varepsilon = 10$	$\varepsilon = 30$	$\varepsilon = 20$
DPMix	0.800	0.782	0.269	0.244
DPMix + Denoising	**0.806**	**0.785**	**0.289**	**0.278**
Local perturbation	0.098	0.098	0.100	0.100
Rand. proj. ($d = 50$)	0.102	0.099	0.099	0.102
Rand. proj. ($d = 100$)	0.102	0.099	0.103	0.100
Rand. proj. ($d = 200$)	0.102	0.101	0.102	0.100
Rand. proj. ($d = 400$)	0.104	0.100	0.100	0.101

IV. Experimental Results

We now compare the performance of DPMix with other data publishing algorithms. Detailed descriptions are deferred to the appendix. The first baseline is local perturbation, which applies the Gaussian mechanism to each data point, i.e., $\ell = 1$. The second baseline is the random projection algorithm. This algorithm first draws a random projection matrix, say $R \in \mathbb{R}^{d \times d_X}$ for some $d \leq d_X$, and transforms each data point by multiplying it with R. It then adds Gaussian noise whose standard deviation is $\sigma(R)(\sqrt{\log(1/(2\delta) + \varepsilon)}/\varepsilon$, where $\sigma(R)$ is the spectral norm of matrix R. It can be shown that this setting yields (ε, δ)-DP [10], [11]. Note that this approach maps the training data into a completely random space, and hence one cannot anymore train a CNN on the released dataset. Thus, we instead train a fully connected network with two hidden layers.

One advantage of differentially private data publishing algorithm is that post-processing of the released dataset does not incur a privacy loss. For instance, one can apply arbitrary post-processing procedures to the synthetic dataset to aid learning. Leveraging this property, we apply Gaussian denoising filters of unit variance prior to training deep learning models. See sample images of denoised mixtures in the appendix.

The performance comparisons are given in Table. II. Notice that our algorithm achieves the best performances, significantly outperforming others. We also observe that the performance of DPMix can be improved by applying denoising before training. On the other hand, the models trained with the datasets generated with the existing algorithms achieve the accuracy of about 0.1, which is the same as that of a random guess.

V. Conclusion

We propose DPMix, a new data publishing algorithm that linearly combines ℓ randomly chosen data points with an additive noise. We provide the differential privacy guarantee of DPMix, showing that it can achieve a reasonably high level of privacy. We also show that one can train both linear and nonlinear models with the dataset generated by DPMix. Particularly, we show that the performance of a deep learning model can be maximized by carefully choosing the value of ℓ. Via extensive experiments, we show that DPMix can achieve significantly improved performances compared to existing data publishing algorithms such as local perturbation and random projection.

ACKNOWLEDGMENT

This work was supported by (1) the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2018R1A1A1A05022889) and (2) the KAIST GCORE (Global Center for Open Research with Enterprise) grant funded by the Ministry of Science and ICT (N11180210).

REFERENCES

[1] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy (SP)*, May 2017.

[2] B. C. Fung, K. Wang, R. Chen, and S. Y. Philip, "Privacy-preserving data publishing: A survey of recent developments," *ACM Computing Surveys*, vol. 42, no. 4, 2010.

[3] T. Zhu, G. Li, W. Zhou, and S. Y. Philip, "Differentially private data publishing and analysis: a survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1619–1638, 2017.

[4] K. Fukuchi, Q. K. Tran, and J. Sakuma, "Differentially private empirical risk minimization with input perturbation," in *International Conference on Discovery Science*. Springer, 2017, pp. 82–90.

[5] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*. Springer Berlin Heidelberg, 2008, pp. 1–19.

[6] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 439–450.

[7] R. Agrawal, R. Srikant, and D. Thomas, "Privacy preserving olap," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. ACM, 2005, pp. 251–262.

[8] S. Agrawal and J. R. Haritsa, "A framework for high-accuracy privacy-preserving mining," in *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. IEEE, 2005, pp. 193–204.

[9] N. Mishra and M. Sandler, "Privacy via pseudorandom sketches," in *Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2006, pp. 143–152.

[10] K. Kenthapadi, A. Korolova, I. Mironov, and N. Mishra, "Privacy via the Johnson-Lindenstrauss Transform," *arXiv preprint*, 2012.

[11] C. Xu, J. Ren, Y. Zhang, Z. Qin, and K. Ren, "DPPro: Differentially private high-dimensional data release via random projection," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 3081–3093, 2017.

[12] X. Xiao, G. Wang, and J. Gehrke, "Differential privacy via wavelet transforms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 8, pp. 1200–1214, 2011.

[13] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 4, p. 25, 2017.

[14] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: Regression analysis under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1364–1375, July 2012.

[15] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *Advances in Neural Information Processing Systems*, 2009, pp. 289–296.

[16] K. Zheng, W. Mou, and L. Wang, "Collect at once, use effectively: Making non-interactive locally private learning possible," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 4130–4139.

[17] N. Agarwal and K. Singh, "The price of differential privacy for online learning," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 32–40.

[18] G. Bernstein, R. McKenna, T. Sun, D. Sheldon, M. Hay, and G. Miklau, "Differentially private learning of undirected graphical models using collective graphical models," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 478–487.

[19] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research (JMLR)*, vol. 12, no. Mar, pp. 1069–1109, Mar. 2011.

[20] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *IEEE Global Conference on Signal and Information Processing*, Dec. 2013, pp. 245–248.

[21] R. Bassily, A. Smith, and A. Thakurta, "Private empirical risk minimization: Efficient algorithms and tight error bounds," in *Proceedings of the 55th Annual Symposium on the Foundations of Computer Science (FOCS '14)*, October 18–21 2014.

[22] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 1069–1109, 2011.

[23] M. Abadi, U. Erlingsson, I. Goodfellow, H. B. McMahan, I. Mironov, N. Papernot, K. Talwar, and L. Zhang, "On the protection of private information in machine learning systems: Two recent approaches," in *IEEE Computer Security Foundations Symposium*, Santa Barbara, CA, Aug. 2017, pp. 1–6.

[24] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, Vienna, Austria, Oct. 2016.

[25] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," in *International Conference on Learning Representations (ICLR)*, Apr. 2017.

[26] C. Karakus, Y. Sun, S. Diggavi, and W. Yin, "Straggler mitigation in distributed optimization through data encoding," in *Advances in Neural Information Processing Systems*, 2017, pp. 5434–5442.

[27] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," in *International Conference on Learning Representations (ICLR)*, Apr. 2018.

[28] ——, "Between-class learning for image classification," *arXiv preprint arXiv:1711.10284*, Nov. 2017.

[29] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations (ICLR)*, Apr. 2018.

[30] H. Inoue, "Data augmentation by pairing samples for images classification," *arXiv:1801.02929*, Jan. 2018.

[31] K. Lee, K. Lee, H. Kim, C. Suh, and K. Ramchandran, "SGD on random mixtures: Private machine learning under data-breach threats," in *ICLR Workshop*, 2018.

[32] I. Mironov, "Rényi differential privacy," in *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*. IEEE, 2017, pp. 263–275.

[33] Y.-X. Wang, B. Balle, and S. Kasiviswanathan, "Subsampled Rényi differential privacy and analytical moments accountant," *arXiv preprint arXiv:1808.00087*, 2018.

[34] Y. LECUN, "The MNIST database of handwritten digits," *http://yann.lecun.com/exdb/mnist/*.

[35] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[36] M. Gil, F. Alajaji, and T. Linder, "Rényi divergence measures for commonly used univariate continuous distributions," *Information Sciences*, vol. 249, pp. 124–131, 2013.

[37] O. Shamir and T. Zhang, "Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes," in *International Conference on International Conference on Machine Learning (ICML)*, Jun. 2013, pp. I–71–I–79.

APPENDIX A
A VARIANT OF DPMIX FOR A LOW PRIVACY REGIME

In a low privacy regime, a slightly modified version of DPMix can achieve an improved privacy guarantee.

Theorem 3 (Privacy guarantee in the low privacy regime): Fix the mixture degree ℓ, the noise level (σ_X, σ_Y) and the number of mixtures T and let $p = \ell/n$. Let $\Delta^2 = \left(\frac{d_X}{\sigma_X^2} + \frac{d_Y}{\sigma_Y^2}\right)$. Then, DPMix($\ell$) is (ε, δ)-DP where

$$\varepsilon = \frac{\alpha^\star Tp}{2\ell^2}\Delta^2 + \frac{\log 1/\delta}{\alpha^\star - 1}, \tag{6}$$

where $\alpha^\star = 1 + \sqrt{\frac{2\ell^2 \log(1/\delta)}{Tp\Delta^2}}$.

Proof: See Section B. ∎

APPENDIX B
PROOFS

A. Proof of Theorem 1

Our theorem is a direct consequence of a recent development in privacy amplification via subsampling [33]. The privacy amplification via subsampling studies the following problem. Consider a mechanism $\mathcal{M}$ that takes ℓ data points. Assume that it is $(\alpha, \varepsilon(\alpha))$-RDP. One can construct a new mechanism $\mathcal{M}'$ that first picks ℓ data points from n data points at random and then applies $\mathcal{M}$ to the chosen data points. Intuitively, the new mechanism $\mathcal{M}'$ is also differentially private, and it is more private than $\mathcal{M}$ due to its subsampling operation. Particularly, the following theorem characterizes the privacy guarantee of the new mechanism $\mathcal{M}'$.

Theorem 4 ([33, Theorem 27]): Let $p = \ell/n$. For all integer $\alpha \geq 2$, $\mathcal{M}'$ is $(\alpha, \varepsilon'(\alpha))$-RDP, where

$$\varepsilon'(\alpha) = \frac{1}{\alpha - 1}\log\Bigg(1 + p^2\binom{\alpha}{2}\min\Big\{4(e^{\varepsilon(2)} - 1), e^{\varepsilon(2)}\min\Big\{2, (e^{\varepsilon(\infty)} - 1)^2\Big\}\Big\} + 4\sum_{j=3}^{\alpha} p^j \binom{\alpha}{j}\sqrt{B(2\lfloor j/2\rfloor)\cdot B(2\lceil j/2\rceil)}\Bigg),$$

where

$$B(\ell) := \sum_{i=0}^{\ell}(-1)^i\binom{\ell}{i}e^{(i-1)\varepsilon(i)}.$$

Indeed, this theorem requires additional conditions on $\mathcal{M}$, and our mechanism satisfies them. Here, we omit those details, referring interested readers to [33, Theorem 27] and [33, Lemma 30]. We now apply this theorem to prove our statement. In our scenario, $\mathcal{M}$ is the mechanism that first computes the average of ℓ data points and then corrupts the average with a multivariate Gaussian noise. In order to apply this theorem, we first need show that this mechanism is $(\alpha, \varepsilon(\alpha))$-RDP for some function $\varepsilon(\cdot)$.

By the definition of RDP, $\varepsilon(\alpha)$ can be found as

$$\varepsilon(\alpha) = \sup_{D \sim D'} D_\alpha(\mathcal{M}(D)||\mathcal{M}(D')). \tag{7}$$

Here, $\mathcal{M}(D) = (X', Y')$, and $X' \sim N(\mu_X, \sigma_X^2 I_{d_X})$ and $Y' \sim N(\mu_Y, \sigma_Y^2 I_{d_Y})$, where μ_X is the average feature value of the ℓ chosen data points and μ_Y is the average label of them. Denote the (μ_X, μ_Y) of dataset D by (μ_X^D, μ_Y^D), and that of D' by $(\mu_X^{D'}, \mu_Y^{D'})$. Since D and D' are adjacent,

$$\|\mu_X^D - \mu_X^{D'}\|^2 \leq \frac{d_X}{\ell^2}, \quad \|\mu_Y^D - \mu_Y^{D'}\|^2 \leq \frac{d_Y}{\ell^2}. \tag{8}$$

Here, the second inequality holds since the label vectors are one-hot-encoded. In [36], the Rényi divergence between multivariate Gaussian distributions $P \sim N(\mu_1, \sigma^2 I_{d_X})$ and $Q \sim N(\mu_2, \sigma^2 I_{d_Y})$ is shown as

$$D_\alpha(P||Q) = \frac{\alpha}{2\sigma^2}\|\mu_1 - \mu_2\|^2. \tag{9}$$

Combining (8) and (9), we have

$$\varepsilon(\alpha) = \frac{\alpha}{2\sigma_X^2}\|\mu_X^D - \mu_X^{D'}\|^2 + \frac{\alpha}{2\sigma_Y^2}\|\mu_Y^D - \mu_Y^{D'}\|^2 = \frac{\alpha\Delta^2}{2\ell^2}. \tag{10}$$

Applying Thm. 4 with (10) concludes the proof.

B. Proof of Theorem 3

Recall in the original DPMix that each mixture contains ℓ data points, which are chosen uniformly at random. Hence, each data point is included in $\frac{T\ell}{n} = Tp$ mixtures on average. We now illustrate a variant of DPMix. In this algorithm, instead of each mixture choosing which data points to mix together, each data point randomly chooses a set of Tp mixtures. Note that this variant behaves very similarly to the original DPMix as n and T tend to infinity.

In this proof, we view the T mixtures as a vector X of length Td_X and a vector Y of length Td_Y. Again, denote the (μ_X, μ_Y) of dataset D by (μ_X^D, μ_Y^D) and that of D' by $(\mu_X^{D'}, \mu_Y^{D'})$. Note the difference in notation compared to the previous proof. Given this notation, the whole data publishing algorithm can be viewed as a Gaussian mechanism. If D and D' are adjacent,

$$\|\mu_X^D - \mu_X^{D'}\|^2 \leq \frac{d_X Tp}{\ell^2}, \quad \|\mu_Y^D - \mu_Y^{D'}\|^2 \leq \frac{d_Y Tp}{\ell^2}. \tag{11}$$

Thus, this together with (9) and (7) gives:

$$\varepsilon(\alpha) = \frac{\alpha Tp}{2\ell^2}\Delta^2. \tag{12}$$

Recall that if a mechanism (α, ε)-RDP, it is $(\varepsilon + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$-DP for any $\delta > 0$. Thus,

$$\varepsilon = \min_{\alpha \geq 1}\left[\frac{\alpha Tp}{2\ell^2}\Delta^2 + \frac{\log 1/\delta}{\alpha - 1}\right]. \tag{13}$$

By optimizing it over α, one can find $\alpha^\star = 1 + \sqrt{\frac{2\ell^2 \log(1/\delta)}{Tp\Delta^2}}$. This concludes the proof.

C. Proof of Theorem 2

We first define the notion of strong convexity.

Definition 1: Let $\mathcal{H}$ be a Hilbert space with inner product $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$. Then, a differentiable function $f : \mathcal{H} \to \mathbb{R}$ is λ-strongly convex if for any w_1 and w_2

$$f(w_1) - f(w_2) \geq \langle \nabla f(w_2), w_1 - w_2 \rangle + \frac{\lambda}{2}\|w_1 - w_2\|^2.$$

We are now ready to provide a more precise theorem statement.

Theorem 5: Assume that the loss function is given by $\mathcal{J}(W) = \frac{1}{2n}\|WX - Y\|_F^2$, and $W^* = \arg\min_W \mathcal{J}(W)$. Consider a variant of SGD in which the computed gradient $g := \nabla\mathcal{L}(W; X', Y')$ is post-processed by a bias correction function $V(g, W) := \ell(g - \sigma_X^2 W)$. By setting the step size as $\alpha_t = \frac{n}{s_{\min}(X)t}$ and assuming $\ell = o(n)$, the output of T^{th} iteration of the variant of SGD satisfies

$$\mathbb{E}[\mathcal{J}(W_T) - \mathcal{J}(W^*)] \leq Q(\sigma_X, \sigma_Y)(1 + \log T)/T, \quad (14)$$

where $Q(\sigma_X, \sigma_Y)$ is increasing both in σ_X and σ_Y.

Proof: We first recall the following theorem, which is a key to proving our convergence guarantees.

Theorem 6 (Thm. 1, [37]): When one applies SGD with a gradient estimator z, the output of the T^{th} iteration satisfies

$$\mathbb{E}[\mathcal{J}(W_T) - \mathcal{J}(W^*)] \leq \frac{17G^2}{\lambda}\left(\frac{1 + \log T}{T}\right), \quad (15)$$

if (1) the gradient estimate is unbiased, i.e., $\mathbb{E}[z] = \nabla_W \mathcal{J}(\cdot)$, (2) $\mathbb{E}[\|z\|^2]$ is bounded by G^2, and (3) $\mathcal{J}(\cdot)$ is λ-strongly convex.

Thus, to prove Thm. 5, it is sufficient to show that the above three conditions hold for our gradient estimator $z = V(g, W)$, and that G is increasing both in σ_X and σ_Y.

(1) Unbiasedness: We first show that $V(g, W)$ is unbiased. Omitting the subscript t, we need to show $\mathbb{E}[V(g, W)] = \nabla_W \mathcal{J}(W, X, Y)$. Equivalently,

$$\mathbb{E}[\nabla_W \mathcal{L}(W, X', Y')] = \frac{1}{\ell}\nabla_W \mathcal{J}(W, X, Y) + \sigma_X^2 W. \quad (16)$$

Since $\nabla_W \mathcal{L}(W, X, Y) = (WX - Y)X^\top$,

$$\begin{aligned}\nabla_W \mathcal{L}(W, X', Y') &= (WX' - Y')(X')^\top \\ &= (W(XC + Q) - (YC + R))(XC + Q)^\top \\ &= (WXC - YC)(XC)^\top + (WQ - R)(XC)^\top \\ &\quad + (WQ - R)Q^\top + (WXC - YC)Q^\top.\end{aligned}$$

We now take the expectation on both sides. Since Q and R are mean zero and independent of C,

$$\begin{aligned}\mathbb{E}[(WQ - R)(XC)^\top] &= \mathbb{E}[WQ - R]\mathbb{E}[(XC)^\top] = 0, \\ \mathbb{E}[(WXC - YC)Q^\top] &= \mathbb{E}[WXC - YC]\mathbb{E}[Q^\top] = 0.\end{aligned}$$

Hence,

$$\begin{aligned}\mathbb{E}[\nabla_W \mathcal{L}(W, X', Y')] &= \mathbb{E}[(WXC - YC) + (XC)^\top] \\ &\quad + \mathbb{E}[(WQ - R)Q^\top].\end{aligned}$$

Here, $\mathbb{E}[(WQ - R)Q^\top] = \mathbb{E}[WQQ^\top] = \sigma_X^2 W$ since Q and R are independent of each other and mean zero, and the covariance matrix of Q is $\sigma_X^2 I_{d_X}$.

Further,

$$\begin{aligned}\mathbb{E}[(WXC - YC)(XC)^\top] &= \mathbb{E}[WXCC^\top X^\top - YCC^\top X^\top] \\ &= WX\mathbb{E}[CC^\top]X^\top - Y\mathbb{E}[CC^\top]X^\top.\end{aligned}$$

Since $\ell = o(n)$, $\mathbb{E}[CC^\top] = \frac{1}{n\ell} I_n$. Thus,

$$\begin{aligned}WX\mathbb{E}[CC^\top]X^\top - Y\mathbb{E}[CC^\top]X^\top &= \frac{1}{n\ell}(WXX^\top - YX^\top) \\ &= \frac{1}{n\ell}(WX - Y)X^\top \\ &= \frac{1}{\ell}\nabla_W \mathcal{J}(W, X, Y).\end{aligned}$$

This proves (16), i.e., $V(g, W)$ is unbiased.

(2) Variance bound: Since

$$\begin{aligned}\mathbb{E}[\|\ell(g - \sigma_X^2 W)\|^2] &\leq \mathbb{E}[\ell^2\|g\|^2 + \ell^2\sigma_X^4\|W\|^2 + 2\ell^2\sigma_X^2\|g\|\|W\|] \\ &\leq \ell^2\mathbb{E}[\|g\|^2] + \ell^2\sigma_X^4\mathbb{E}[\|W\|^2] \\ &\quad + 2\ell^2\sigma_X^2\mathbb{E}[\|g\|]\mathbb{E}[\|W\|],\end{aligned}$$

it is sufficient to show that $\mathbb{E}[\|g\|]$ and $\mathbb{E}[\|g\|^2]$ are bounded above by a function that is increasing both in σ_X and σ_Y. By using the triangle inequality, we have

$$\begin{aligned}\mathbb{E}[\|g\|] &= \mathbb{E}[\|(W(XC + Q) - (YC + R))(XC + Q)^\top\|] \\ &\leq \underbrace{\mathbb{E}[\|W(XC + Q)(XC + Q)^\top\|]}_{\text{Increasing in } \sigma_X} \\ &\quad + \underbrace{\mathbb{E}[\|(YC + R)(XC + Q)^\top\|]}_{\text{Increasing in } \sigma_X \text{ and } \sigma_Y},\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}[\|g\|^2] &\leq \underbrace{\mathbb{E}[\|W(XC + Q)(XC + Q)^\top\|^2]}_{\text{Increasing in } \sigma_X} \\ &\quad + \underbrace{\mathbb{E}[\|(YC + R)(XC + Q)^\top\|^2]}_{\text{Increasing in } \sigma_X \text{ and } \sigma_Y} \\ &\quad + 2\underbrace{\mathbb{E}[\|W(XC + Q)(XC + Q)^\top\|]}_{\text{Increasing in } \sigma_X} \\ &\quad \cdot \underbrace{\mathbb{E}[\|(YC + R)(XC + Q)^\top\|]}_{\text{Increasing in } \sigma_X \text{ and } \sigma_Y}.\end{aligned}$$

This proves that $\mathbb{E}[\|z\|^2]$ is bounded above by a function that is increasing both in σ_X and σ_Y.

(3) Strong convexity: We now show the last condition, i.e., $\mathcal{J}(\cdot)$ is strongly convex. Given any W_1 and W_2,

$$\begin{aligned}\mathcal{J}(W_1) - \mathcal{J}(W_2) &= \frac{1}{2n}\|W_1X\|_F^2 - \frac{1}{2n}\|W_2X\|_F^2 \\ &\quad - \frac{1}{n}\langle(W_1 - W_2)X, Y\rangle. \quad (17)\end{aligned}$$

Using $\nabla_W \mathcal{J}(W) = \frac{1}{n}(WX - Y)X^\top$, one can show that

$$\begin{aligned}&\mathcal{J}(W_1) - \mathcal{J}(W_2) - \langle\nabla_W \mathcal{J}(W_1), W_1 - W_2\rangle \\ &= \frac{1}{2n}\|(W_1 - W_2)X\|_F^2 \\ &\geq \frac{s_{\min}(X)}{2n}\|W_1 - W_2\|_F^2.\end{aligned}$$

By definition, $\mathcal{J}(\cdot)$ is λ-strongly convex with $\lambda = \frac{s_{\min}(X)}{n}$. ∎

APPENDIX C
DETAILS ABOUT EXPERIMENTAL SETUPS

A. Training setup

The details about CNN architecture for DPMix and a fully connected network for random projection algorithm are as follows.

- (CNN) Input is a 32 x 32 x 3 image that is passed through 2 convolutional layers followed by 3 fully connected layers, the last one encoding the 10-dimensional class vector: (1) Conv5x5, 32 feature maps, (2) MaxPool2x2, stride = 2, (3) Conv5x5, 48 feature maps, (4) MaxPool2x2, stride = 2, (5) FC100, (6) FC100, (7) FC10.
- (FC) Input is a randomly projected vector that is passed through 3 fully connected layers, the last one encoding the 10-dimensional class vector: (1) FC100, (2) FC100, (3) FC10.

The training loss is optimized by AdamOptimizer with the default hyperparameters. We use an initial learning rate of 0.0001 and halve the learning rate every 10 epochs. We use mini-batches of size 8. Note that this is *different* from the mixture degree ℓ. We run the training algorithm for 50 epochs. Assuming that a small fraction of real data is available for the purpose of validation, we choose the best performing epoch based on the validation accuracy.

B. Sample images of denoised mixtures

In Section IV, we observed that applying a Gaussian denoising filter prior to training can improve the prediction accuracies. Here, we visualize sample images of denoised mixtures in Fig. 4.

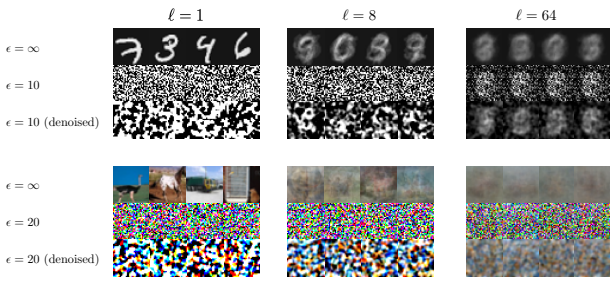


Fig. 4: Sample images of denoised mixtures.