Differentially Private Kernel Inducing Points using features from ScatterNets (DP-KIP-ScatterNet) for Privacy Preserving Data Distillation

Margarita Vinaroz *margarita.vinaroz@tuebingen.mpg.de*
University of Tübingen
International Max Planck Research School for Intelligent Systems (IMPRS-IS)

Mi Jung Park *mijungp@cs.ubc.ca, mjupa@dtu.dk*
University of British Columbia & Alberta Machine Intelligence Institute (AMII)
Technical University of Denmark

Reviewed on OpenReview: *https://openreview.net/forum?id=84M8xwNxrc*

Abstract

Data distillation aims to generate a small data set that closely mimics the performance of a given learning algorithm on the original data set. The distilled dataset is hence useful to simplify the training process thanks to its small data size. However, distilled data samples are not necessarily privacy-preserving, even if they are generally humanly indiscernible. To address this limitation, we introduce differentially private kernel inducing points (DP-KIP) for privacy-preserving data distillation. Unlike our original intention to simply apply DP-SGD to the framework of KIP, we find that KIP using infinitely-wide convolutional neural tangent kernels (conv-NTKs) performs better compared to KIP using fully-connected NTKs. However, KIP with conv-NTKs, due to its convolutional and pooling operations, introduces an unbearable computational complexity, requiring hundreds of V100 GPUs in parallel to train, which is impractical and more importantly, such computational resources are inaccessible to many. To overcome this issue, we propose an alternative that does not require pre-training (to avoid a privacy loss) and can well capture complex information on images, as those features from conv-NKTs do, while the computational cost is manageable by a single V100 GPU. To this end, we propose DP-KIP-ScatterNet, which uses the wavelet features from *Scattering networks* (ScatterNet) instead of those from conv-NTKs, to perform DP-KIP at a reasonable computational cost. We implement *DP-KIP-ScatterNet* in – computationally efficient – JAX and test on several popular image datasets to show its efficacy and its superior performance compared to state-of-the art methods in image data distillation with differential privacy guarantees. Our code is available at `https://github.com/ParkLabML/DP-KIP`.

1 Introduction

First introduced by Wang et al. (2018), data distillation (DD) aims at extracting the knowledge of the entire training dataset to a few synthetic, distilled datapoints. What DD offers is that the models trained on the small number of distilled datapoints achieve high-performance relative to the models trained on the original, large training dataset. DD is a sensible choice for fast, cheaper, and light training of neural network models. Various applications of DD include continual learning (Liu et al., 2020; Rosasco et al., 2021; Sangermano et al., 2022; Wiewel & Yang, 2021; Masarczyk & Tautkute, 2020), neural architecture search (Zhao & Bilen, 2021; Zhao et al., 2021; Zhao & Bilen, 2023), and more.

Depending on the similarity metrics chosen for judging how close the small distilled datasets are to the original large datasets, there are different ways to formulate the DD problem. For instance, Zhao et al. (2021)

formulate it as a gradient matching problem between the gradients of deep neural network weights trained on the original and distilled data. Nguyen et al. (2021a;b) formulate it as a kernel ridge regression problem where the distilled data correspond to the kernel inducing points (KIP). Regardless of what formulation one takes, the techniques for DD are fast improving, and their application domains are widening.

Among the many application domains, Nguyen et al. (2021a) claim that DD is also useful for privacy-preserving dataset creation, by showing distilled images with 90% of their pixels corrupted while test accuracy with those exhibits limited degradation. It is true that the distilled images with 90% of corrupted pixels are not humanly discernible. However, their illustration is merely experimental and does not involve any formal definition of privacy. Recently, Dong et al. (2022) attempt to connect DD with differential privacy (Dwork & Roth, 2014), one of the popular privacy notions, based on DD's empirical robustness against some known attacks. Unfortunately, the empirical evaluation of the method and its theoretical analysis contain major flaws, as described in Carlini et al. (2022).

For a provable privacy guarantee, Chen et al. (2022) apply *DP-SGD*, the off-the-shelf differential privacy algorithm (Abadi et al., 2016), to optimizing a gradient matching objective to estimate a differentially private distilled dataset. More recently, Zheng & Li (2023) proposes a differentially private distribution matching framework, which further improves the performance of Chen et al. (2022).

In this paper, we apply DP-SGD to the KIP framework developed by Nguyen et al. (2021a;b) for differentially private data distillation. There are two important reasons we choose to privatize KIP over other existing DD algorithms. First, in DP-KIP, the gradients that DP-SGD privatize are the distilled datapoints. Typically, we consider only a few distilled datapoints and therefore *the dimension of the parameters to guard is relatively low*. Consequently, the privacy-accuracy trade-off of DP-KIP is better than that of the gradient matching framework with DP-SGD (Chen et al., 2022), as the latter needs to privatize significantly higher dimensional neural network parameters than the former. Second, optimizing for distilled data under KIP is *computationally cheaper and converges faster* than that under gradient matching. KIP implements kernel ridge regression (KRR) which involves convex optimization with a closed-form solution and estimating distilled data requires first-order optimization methods, which is straightforward to apply DP-SGD to and converges relatively fast. However, the gradient matching framework requires computationally expensive bi-level optimization (inner loop updates distilled data and outer loop updates the neural network parameters), which requires a longer training time than KIP. In non-DP settings, a longer training time and a higher parameter dimension typically do not matter. However, in DP settings, these incur higher privacy costs, which in turn results in a poorer privacy-utility trade-off. Indeed, for these reasons, we observe that empirically our method outperforms other DP-DD methods at the same privacy level.

Settling with the kernel-based KIP framework, now the question is which features (kernel) to use to produce high quality distilled datasets with privacy guarantees. Nguyen et al. (2021a) use the features of infinitely-wide neural tangent kernel (NTK) from fully connected neural networks. Nguyen et al. (2021b) significantly improves previous results in Nguyen et al. (2021a) by changing the infinite-width NTK from a fully connected network to a convolutional network. However, kernels formed out of convolutional and pooling layers introduce a significant computational burden, due to the necessity of keeping track of pixel-pixel correlations. To mitigate this problem, Nguyen et al. (2021b) use hundreds of V100 GPUs working in parallel to make the optimization computationally feasible. As a result, using the infinitely-wide ConvNet NTK (ConvNet-NTK) becomes out of reach for researchers who do not have hundred V100 GPUs available at hand. So, we question which features (kernels) to use to produce the results better or comparable to the distilled data trained with the features of infinitely-wide ConvNet NTKs, while keeping the computational cost manageable to a single V100 GPU?

The first alternative we consider is an empirical NTK (e-NTK), a finite-dimensional approximation to the infinite-width NTK. e-NTK is known to provide useful approximations of the training dynamics and generalization properties of a given deep neural network (Schoenholz et al., 2017; Xiao et al., 2018; 2020). It has been observed that as the width of the neural network increases, the e-NTK at initialization approaches the infinite-width NTK. Therefore, we empirically test whether the performance of convolutional e-NTKs is comparable to that obtained by the convolutional infinitely-width NTK.

The second alternative we consider is a kernel using the features from a *Scattering Network*, often called ScatterNet (Oyallon & Mallat, 2015). ScatterNet is a feature extractor formed by the combination of predefined complex wavelet filters and non-linear operations followed by a final averaging operator. These handcrafted features exhibit stability to deformations and invariance to translations, making them a suitable choice for image classification tasks such as DD, where the goal is to learn distilled images that minimize the L2-norm between true and predicted labels. Notably, Tramer & Boneh (2021) previously studied the efficacy of these features in image *classification tasks* with differential privacy guarantees. We attempt to use ScatterNet features for data distillation with differential privacy guarantees. We use an untrained ScatterNet (i.e., at its initialization) to avoid privacy costs for pre-training the network.

The last alternative we consider is a kernel using *perceptual features* (PFs). PFs are defined as the concatenation of a (pre-trained) deep neural network activations for a given input. They have been widely used in different tasks such as transfer learning (Yosinski et al., 2014a; Huh et al., 2016), super-resolution (Johnson et al., 2016) or image generation (Santos et al., 2019; Harder et al., 2023). Although DD task is different from those mentioned beforehand, PFs have been shown to extract useful information from input images and therefore we investigate if the use of these features can contribute to improving the performance of KIP. We use an untrained ResNet-18 (i.e., at its initialization) to avoid privacy costs for pre-training the network.

We show that KIP with ScatterNet features outperforms e-NTK features and perceptual features, achieving similar performance to the infinite-width convolutional NTK, in non-DP settings. Therefore, we incorporate ScatterNet features in DP-KIP. To separate ours from the original KIP, we call our algorithm by *DP-KIP-ScatterNet*. Our experiments show that DP-KIP-ScatterNet significantly outperforms DP-gradient matching by Chen et al. (2022) and DP-distribution matching by Zheng & Li (2023), when tested on several benchmark image classification datasets.

Before moving on to describe our method, we summarize our contributions

- We propose a DP data distillation framework based on KIP, which uses DP-SGD for privacy guarantees in the resulting distilled data. This itself is a mere application of DP-SGD to an existing data distillation method. However, motivated by the unbearable computational costs in using the infinite-width convolutional NTKs, we look for alternative features and empirically observe that the features from ScatterNets are the most useful in distilling image datasets, evaluated on classification tasks. As a result, DP-KIP-ScatterNet significantly outperforms state-of-the-art DP data distillation methods.
- We further improve the performance of DP-KIP-ScatterNet by considering varying levels of pixel corruption rates. By corrupting some portion of pixels with some noise, we reduce the dimension of distilled data, which results in better privacy-accuracy trade-offs.
- Unlike other most existing data distillation papers that focus on distilling *image data only*, we also test DP-KIP, where we use the original features[1] in KIP, i.e., the infinitely-wide fully-connected NTKs, on distilling *tabular datasets*. DP-KIP with a relatively small number of distilled samples (only 10 samples/class) significantly outperforms previous DP data generation models at the same privacy level.

2 Background

In the following section, we review the Kernel Inducing Points (KIP) algorithm, different kernel functions we propose to use in KIP (infinite-width NTK, e-NTK, ScatterNet features, PFs) and differential privacy (DP).

2.1 KIP

We start giving some key definitions and describing KIP algorithm from Nguyen et al. (2021a).

[1]For this task, we use the original inifinite-width fully connected NTKs since tabular data would not benefit from those features engineered for image data such as PFs and ScatterNet features.

Definition 2.1. *Fix a loss function l and let $f, \tilde{f} : \mathbb{R}^D \to \mathbb{R}^C$ be two functions. Let $\xi \geq 0$. Given a distribution $\mathcal{P}$ on $\mathbb{R}^D \times \mathbb{R}^C$, we say f and $\tilde{f}$ are weakly ξ-close with respect to $(l, \mathcal{P})$ if:*

$$\left|\mathbb{E}_{(x,y)\in\mathcal{P}} l(f(x), y) - \mathbb{E}_{(x,y)\in\mathcal{P}} l(\tilde{f}(x), y)\right| \leq \xi. \tag{1}$$

Definition 2.2. *Let $\mathcal{D}$ and $\tilde{\mathcal{D}}$ be two labeled datasets in $\mathbb{R}^D$ with label space $\mathbb{R}^C$ and $A, \tilde{A}$ be two fixed learning algorithms. Let $\xi \geq 0$. Given the resulting model $A_{\mathcal{D}}$ obtained after training A on $\mathcal{D}$, the resulting model $\tilde{A}_{\tilde{\mathcal{D}}}$ obtained after training $\tilde{A}$ on $\tilde{\mathcal{D}}$, we say $\tilde{\mathcal{D}}$ is a ξ-approximation of $\mathcal{D}$ with respect to $(\tilde{A}, A, l, \mathcal{P})$ if $\tilde{A}_{\tilde{\mathcal{D}}}$ and $A_{\mathcal{D}}$ are weakly ξ-close with respect to $(l, \mathcal{P})$, where l is a loss function and $\mathcal{P}$ is a distribution on $\mathbb{R}^D \times \mathbb{R}^C$.*

In data distillation, the goal is to find a small dataset $\mathcal{D}_s$ that is ξ-approximation to a large, original dataset $\mathcal{D}_t$ drawn from a distribution $\mathcal{P}$ with respect to a learning algorithm A and a loss function l:

$$\left|\mathbb{E}_{(x,y)\in\mathcal{P}} l(A_{\mathcal{D}_t}(x), y) - \mathbb{E}_{(x,y)\in\mathcal{D}_t} l(A_{\mathcal{D}_s}(x), y)\right| \leq \xi. \tag{2}$$

In KIP, the loss l is a classification accuracy in terms of the L2-distance between true labels and predicted labels; and the learning algorithm A is kernel ridge regression (KRR).

Consider a target dataset $\mathcal{D}_t = \{(\mathbf{x}_{t_i}, y_{t_i})\}_{i=1}^n$ with input features $\mathbf{x}_{t_i} \in \mathbb{R}^D$ and scalar labels y_{t_i}. Given a kernel k , the KIP algorithm constructs a small distilled dataset $\mathcal{D}_s = \{(\mathbf{x}_{s_j}, y_{s_j})\}_{j=1}^m$, where $\mathbf{x}_{s_j} \in \mathbb{R}^D$, and scalar labels y_{s_j} and importantly $m \ll n$, such that its performance on a classification task approximates the performance of the target dataset for the same task. Note that Nguyen et al. (2021a) call $\mathcal{D}_s$ "support" dataset (hence the subscript "s"). In this paper, we will use "support" and "distilled" datasets, interchangeably.

The KIP algorithm, we start by randomly initializing the support dataset and then iteratively refine $\mathcal{D}_s$ by minimizing the Kernel Ridge Regression (KRR) loss:

$$\mathcal{L}(\mathcal{D}_s) = \sum_{i=1}^{n} (y_{t_i} - \mathbf{k}_{t_i s}{}^\top (K_{ss} + \lambda I)^{-1} \mathbf{y}_s)^2, \tag{3}$$

with respect to the support dataset $\mathcal{D}_s$. Here $\lambda > 0$ is a regularization parameter, K_{ss} is a kernel matrix, where the (i, j)-th entry is $k(\mathbf{x}_{s_i}, \mathbf{x}_{s_j})$, $\mathbf{k}_{t_i s}$ is a column vector where the j-th entry is $k(\mathbf{x}_{t_i}, \mathbf{x}_{s_j})$, and $\mathbf{y}_s$ is a column vector where the j-th entry is y_{s_j}. During the training phase, the support dataset is updated using a gradient-based optimization method, e.g., using Stochastic Gradient Descent (SGD), until some convergence criterion is satisfied.

2.2 Infinite-width NTK

Initially, NTKs were proposed to help understand neural networks' training dynamics in a function space. In particular, in the infinite-width limit, the parameters of a neural network do not change from the random initialization over the course of the training and the gradients of the network parameters converge to an infinite-dimensional feature map of the NTK Jacot et al. (2018); Lee et al. (2019); Arora et al. (2019); Lee et al. (2020). Characterizing this neural tangent kernel is essential to analyzing the convergence of training and generalization of the neural network.

Based on this finding, Nguyen et al. (2021a) motivate the use of NTK in KIP in the following sense: (a) the kernel ridge regression with an NTK approximates the training of the corresponding infinitely-wide neural network; and (b) it is *likely* that the use of NTK yields approximating $\mathcal{D}_t$ by $\mathcal{D}_s$ (in the sense of ξ-approximations given in eq. 2) for learning algorithms given by a broad class of neural networks. While there is no mathematical proof on point (b), Nguyen et al. (2021a) empirically backed up point (b).

2.3 Empirical NTK (e-NTK)

The empirical NTK (e-NTK) is a finite-dimensional approximation of the infinite-width NTK that provides a way to approximate the dynamics of deep neural networks during training. The e-NTK for a given (finite wide) neural network f, with random initial parameters θ is defined as:

$$\text{e-NTK}(\mathbf{x}, \mathbf{x}') = \left[\frac{\partial f(\theta, \mathbf{x})}{\partial \theta}\right] \left[\frac{\partial f(\theta, \mathbf{x}')}{\partial \theta}\right]^\top \tag{4}$$

where $\partial f(\theta, \cdot)/\partial \theta$ denotes the neural network Jacobian. In our experiments, we considered the e-NTK obtained by using a Lenet (Lecun et al., 1998) and ResNet18 (He et al., 2016) architectures for grayscale and RGB images respectively.

2.4 ScatterNet features

We consider the kernel defined by the inner product of the Scattering Network (ScatterNet) features presented by Oyallon & Mallat (2015). ScatterNet is a Scale Invariant Feature Transform (SIFT) feature extractor based on cascades of wavelet transform convolutions followed by non-linear modulus and a final averaging operator. For a given input image $\mathbf{x}$, its ScatterNet feature representation is defined as the output of the Scattering Network of depth J:

$$\phi_S(\mathbf{x}) := A|W_2|W_1 x|| \tag{5}$$

where W_2 and W_1 are wavelet transforms that capture local and global information of the input image, $|\cdot|$ is the modulus operator which computes the magnitude of each wavelet coefficient and guarantees translation-invariant representations, and A is the 2^J patch-averaging operator that contributes in capturing higher-level patterns and relationships between different parts of the image. The ScatterNet can be seen as a variant of a Convolutional Neural Network (CNN) where the architecture parameters and filters are not learned during the training phase, but are predefined fixed wavelets transforms.

Following Oyallon & Mallat (2015), we use a ScatterNet of depth $J = 2$ with wavelets rotated along eight angles. The extracted ScatterNet features for a given image of size $H \times W$ has dimension $(K, \frac{H}{2^J}, \frac{W}{2^J})$ where K is set to 81 and 243 for grayscale and RBG images respectively.

2.5 Perceptual features (PFs)

Here we propose the kernel defined by the inner product of the features extracted from deep convolutional networks (DCNNs), also known as perceptual features (PFs). These features are defined as the concatenation of each layer's output from a fixed deep convolutional neural network.

In this work we considered the features extracted form a randomly initialized ResNet18 (He et al., 2016).

2.6 Differential privacy (DP)

Differential privacy is a gold standard privacy notion in machine learning and statistics. Its popularity is due to the mathematical probability. The definition of DP (Definition 2.4 in Dwork & Roth (2014)) is given below.

Definition 2.3. *A randomized mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private if for all neighboring datasets $\mathcal{D}, \mathcal{D}'$ differing in an only single entry and all sets $S \subset range(\mathcal{M})$, the following inequality holds:*

$$Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^\epsilon \cdot Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta$$

The definition states that for all datasets differing in an only single entry, the amount of information revealed by a randomized algorithm about any individual's participation is bounded by ϵ and δ (which is preferably smaller than $1/|\mathcal{D}|$). In our work, we use the inclusion/exclusion definition for neighbouring datasets and privacy per item or per (training) example as protection granularity.

A common paradigm for constructing differentially private algorithms is to add calibrated noise to an algorithm's output. In our algorithm, we use the *Gaussian mechanism* to ensure that the distilled dataset satisfies the DP guarantee. For a deterministic function $h : \mathcal{D} \to \mathbb{R}^d$, the Gaussian mechanism is defined by

$\tilde{h}(\mathcal{D}) = h(\mathcal{D}) + \mathcal{N}(0, \Delta_h^2 \sigma^2 I_d)$. Here, the noise scale depends on the *global sensitivity* (Dwork et al., 2006a) of the function h, Δ_h, and it is defined as the maximum difference in terms of l_2-norm, $\|h(\mathcal{D}) - h(\mathcal{D}')\|_2$, for $\mathcal{D}$ and $\mathcal{D}'$ differing in an only single entry and σ is a function of the privacy level parameters, ϵ, δ.

Differential privacy has two fundamental properties that are useful for applications like ours: composability (Dwork et al., 2006a) and post-processing invariance (Dwork et al., 2006b). Composability ensures that if all components of a mechanism are differentially private, then its composition is also differentially private with some privacy guarantee degradation due to the repeated use of the training data. In our algorithm, we use the subsampled RDP composition by (Wang et al., 2019), as they yield to tight bounds for the RDP parameter, ϵ_{RDP}, for a subsampled mechanism. To switch from the definition of RDP to (ϵ, δ)-DP, we follow (Wang et al., 2019), which keeps track of the parameters of RDP and converts them to the definition of (ϵ, δ)-DP through the use of numerical methods. Furthermore, the post-processing invariance property states that any application of arbitrary data-independent transformations to an (ϵ, δ)-DP algorithm is also (ϵ, δ)-DP. In our context, this means that no information other than the allowed by the privacy level ϵ, δ, can be inferred about the training data from the privatized mechanism.

3 Our algorithm: DP-KIP and DP-KIP-ScatterNet

In this section, we introduce our proposed algorithm *differentially private kernel inducing points (DP-KIP)*. The algorithm produces differentially private distilled samples by clipping and adding calibrated noise to the distilled data's gradients during training. *DP-KIP-ScatterNet* is a particular algorithm of DP-KIP which uses the features from Scattering Networks.

3.1 Outline of DP-KIP

Our algorithm is shown in Algorithm 1. We first initialize the distilled (support) dataset $\mathcal{D}_s$, where the learnable parameters $\mathbf{X}_s = \{\mathbf{x}_{s_j}\}_{j=1}^m$ are drawn from a standard Gaussian distribution, (i.e. $\mathbf{x}_{s_j} \sim \mathcal{N}(0, I_D)$ for $\mathbf{x}_{s_j} \in \mathbb{R}^D$). We generate labels $\mathbf{y}_s$ by drawing them from a uniform distribution over the number of classes; and fix them during the training for $\mathbf{X}_s$. Note that the original KIP algorithm has an option for optimizing the labels through *Label Solve* given optimized distilled images. However, we choose not to optimize for the labels to reduce the privacy loss incurring during training.

At each iteration of the algorithm, we randomly subsample B samples from the target dataset $\mathcal{D}_t$, $\mathcal{D}_{t_B}$. Given a kernel k, we compute the loss given in eq. 3. Then, we compute the per target sample gradients with respect to the support dataset, given in

$$\begin{aligned} g(\mathbf{x}_{t,l}, y_{t,l}) &:= \nabla_{\mathcal{D}_s} \mathcal{L}(\mathcal{D}_s) \\ &= \nabla_{\mathcal{D}_s} \left(\mathbf{y}_{t_l} - \mathbf{k}_{t_l s} (K_{ss} + \lambda I)^{-1} \mathbf{y}_s \right)^2 . \end{aligned} \tag{6}$$

As in DP-SGD, we ensure each datapoint's gradient norm is bounded by explicitly normalizing the gradients if its $l2$-norm exceeds C. In the last steps of the algorithm, the clipped gradients are perturbed by the Gaussian mechanism and averaged as in DP-SGD algorithm and finally, the support dataset is updated by using some gradient-based method (e.g. SGD, Adam).

Prop. 1 states that the proposed algorithm is differentially private.

Proposition 1. *The DP-KIP algorithm produces a (ϵ, δ)-DP distilled dataset.*

Proof. Due to the Gaussian mechanism, the noisy-clipped gradients per sample are DP. By the post-processing invariance property of DP, the average of the noisy-clipped gradients is also DP. Finally, updating the support dataset with the aggregated-noisy gradients and composing through iterations with the subsampled RDP composition (Wang et al., 2019) produces (ϵ, δ)-DP distilled dataset. The exact relationship between (ϵ, δ), T (number of iterations DP-KIP runs), B (mini-batch size), N (number of datapoints in the target dataset), and σ (the privacy parameter) follows the analysis of Wang et al. (2019). □

Algorithm 1 DP-KIP

Input: Dataset $\mathcal{D}_t = \{(\mathbf{x}_{t_i}, y_{t_i})\}_{i=1}^n$, number of distilled samples to generate m, number of iterations P, mini-batch size B, clipping norm C, privacy level (ϵ, δ)
Step 1. Initialize distilled dataset $\mathcal{D}_s = \{(\mathbf{x}_{s_j}, y_{s_j})\}_{j=1}^m$ with $\mathbf{x}_{s_i} \sim \mathcal{N}(0, I_D)$
Step 2. Given a desired level of (ϵ, δ)-DP, we compute the privacy parameter σ using the auto-dp package by Wang et al. (2019).
for $p = 1$ **to** P **do**
  Step 3. Randomly subsample $\mathcal{D}_{t_B} = \{(\mathbf{X}_{t_B}, \mathbf{y}_{t_B})\}$
  Step 4. Compute KRR loss given in eq. 3.
  Step 5. Compute per-sample gradients in eq. 6 for each $l \in t_B$.
  Step 6. Clip the gradients via $\hat{g}(\mathbf{x}_l) = g(\mathbf{x}_l) / \max(1, \|g(\mathbf{x}_l)\|_2 / C)$
  Step 7. Add noise: $\tilde{g} = \sum_{l=1}^B \hat{g}(\mathbf{x}_l) + \mathcal{N}(0, \sigma^2 C^2 I)$.
  Step 8. Update distilled dataset $\mathcal{D}_s$ with SGD.
end for
Return: Learned private support dataset $\mathcal{D}_s$

3.2 A few thoughts on the algorithm

Support dataset initialization: The first step in DP-KIP initializes each support datapoint in $\mathcal{D}_s$ to be drawn from the standard Gaussian distribution, $\mathcal{N}(0, I_D)$. This random initialization ensures that no sensitive information is inherit by the algorithm at the beginning of training. Nevertheless, one can choose a different type of initialization such as randomly selecting images from the training set as in (Nguyen et al., 2021a;b) and then, privatize those to ensure that no privacy violation incurs during the training process. The downside of this approach is that the additional private step in initialization itself is challenging, since computing the sensitivity for neighboring datasets has no trivial bound on the target dataset and incurs in an extra privacy cost.

Clipping effect of the gradients: In our algorithm, we follow the approach from (Abadi et al., 2016) and clip the gradients to have l_2-norm C. This clipping norm is treated as an hyperparameter since gradient values domain is unbounded a priori. Setting C to a relatively small value is beneficial during training as the total noise variance is scaled down by the C factor. However, the small value may result in a large amount of the gradients being clipped and thus, drastically discard useful or important information. In contrast, setting C to a large value helps preserving more information encoded in the gradients but it yields to a higher noise variance being added to the gradients, which may worsen the learned results. Consequently, finding a suitable C is crucial to maintain a good privacy-accuracy trade-off on DP-KIP algorithm. See Sec. A in Appendix for experimental results of the clipping effect.

Regularization parameter: Following Nguyen et al. (2021a) we set the regularization parameter λ in eq. 3 to $\frac{1}{m}\lambda \cdot \mathrm{tr}(K_{ss})$ where m is the number of datapoints in the distilled dataset X_s. In this fashion, the regularization parameter is scale-invariant to the kernel function and ensures a similar performance of the algorithm for a wide range of λ values.

Why not privatize the feature maps? Since in this work we are considering kernel functions k with finite dimensional feature maps ϕ, such that $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$ and the only data dependent term in KIP is $k_{t_i s}$ in eq. 3, one could consider privatizing $\phi(\mathbf{x}_{t_i}), \forall i \in [n]$ once at the beginning and reuse them during the training of the algorithm. However, perturbing the feature maps results in a constant signal-to-noise ratio, inversely proportional to the parameter noise σ, regardless of the number of datapoints in $\mathcal{D}_t$. This observation implies that when the feature maps are privatized, they essentially become dominated by noise. See Sec. B in Appendix for a detailed explanation.

4 Related Work

The most closely related work is Chen et al. (2022), which applies *DP-SGD* on the gradient matching objective. As concurrent work, Zheng & Li (2023) proposes a differentially private distribution matching framework.

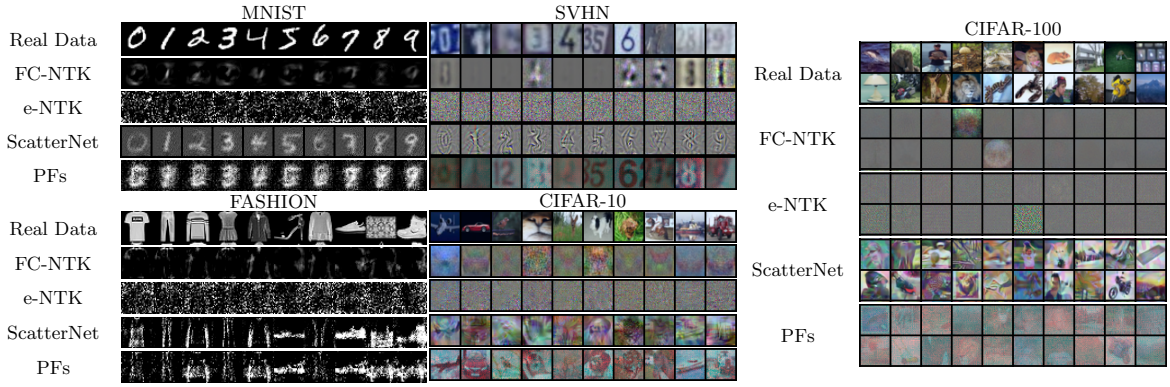

Figure 1: Generated image samples from KIP comparison models.

Our work differs from these two in the following sense. First, we use kernel functions as features in comparing the distilled and original data distributions. Second, we formulate our problem as kernel inducing points, our DP-SGD's privacy-accuracy trade-off is better than that of gradient matching due to privatizing a smaller dimensional quantity in our case.

Another line of relevant work is differentially private data generation. While there are numerous papers to cite, we focus on a few that we compare our method against in Sec. 5. The majority of existing work uses DP-SGD to privatize a generator in the Generative Adversarial Networks (GANs) framework. In GANs, the generator never has direct access to training data and thus requires no privatization, as long as the discriminator is differentially private. Examples of this approach include DP-CGAN (Torkzadehmahani et al., 2019), DP-GAN (Xie et al., 2018), G-PATE (Long et al., 2021), DataLens (Wang et al., 2021), and GS-WGAN Chen et al. (2020). A couple of recent work outside the GANs framework include DP-MERF Harder et al. (2021), DP-HP (Vinaroz et al., 2022), DP-NTK (Yang et al., 2023) and DP-MEPF (Harder et al., 2023) that use the approximate versions of maximum mean discrepancy (MMD) as a loss function; and DP-Sinkhorn (Cao et al., 2021) that proposes using the Sinkhorn divergence in privacy settings. These data generation methods aim for more general-purpose machine learning. On the contrary, our method aims to create a privacy-preserving small dataset that matches the performance of a particular task in mind (classification). Hence, when comparing them in terms of classification performance, it is not necessarily fair for these general data generation methods. Nevertheless, at the same privacy level, we still want to show where our method stands relative to these general methods in Sec. 5.

In the line of work of dataset reconstruction attacks, Loo et al. (2023) propose a reconstruction attack for trained neural networks that is a generalization of KIP. In addition to the task of reconstructing original training samples, they show that their attack can be used for dataset distillation when the number of samples to reconstruct is set to be significantly smaller than that of the original dataset. While the distilled samples from the reconstruction attack do not resemble any original image from the training set, there is no formal privacy guarantee about the generated images.

5 Experiments

Here, we show the performance of KIP and DP-KIP over different real world datasets. In Sec. 5.1 we follow previous data distillation work and focus our study on grayscale and color image datasets. In addition, we also test DP-KIP performance on imbalanced tabular datasets with numerical and categorical features in Sec. 5.2. All experiments were implemented using JAX (Bradbury et al., 2018), except KIP e-NTK experiments were we used `autograd.grad` function implemented in PyTorch (Paszke et al., 2019). All the experiments were run on a single NVIDIA V100 GPU.

5.1 Image data

We start by testing KIP and DP-KIP performance on MNIST (LeCun et al., 2010), FashionMNIST (Xiao et al., 2017), SVHN (Netzer et al., 2011), CIFAR-10 (Krizhevsky & Hinton, 2009) and CIFAR-100 datasets for image classification.

Each of MNIST and FashionMNIST datasets consists of 60,000 samples of 28×28 grey scale images depicting handwritten digits and items of clothing, respectively, sorted into 10 classes. The SVHN dataset contains 600,000 samples of 32×32 colour images of house numbers taken from Google Street View images, sorted into 10 classes. The CIFAR-10 dataset contains a 50,000-sample of 32×32 colour images of 10 classes of objects, such as vehicles (cars and ships) and animals (horses and birds). CIFAR100 contains 100 classes of objects.

In Table 1, we consider KIP with infinite-width NTK for fully connected (KIP FC-NTK) and convolutional (KIP ConvNet-NTK) networks as a non-private baseline. We show as evaluation metric the averaged test accuracy using KRR as a classifier for 1, 10 and 50 distilled images per class over 5 independent runs. The general trend for different KIP variants is that creating more images per class improves the classification accuracy. The proposed e-NTK, ScatterNet and PFs features increase significantly the accuracy with respect to KIP FC-NTK for all the considered image datasets and more interestingly, KIP ScatterNet outperforms or performs similar to KIP ConvNet-NTK for most datasets. Fig. 1 shows learned distilled images obtained by the different considered kernels. Both, infinite and empirical NTKs distilled images look blurrier and are visually more indistinguishable between the different classes and datasets. On the contrary, ScatterNet and PFs distilled images are visually more interpretable.

Our intuition for the interpretability of the images distilled from PFs relies on the information carried by the DCNN layer's outputs detailed in Yosinski et al. (2014b). DCNNs layer's outputs transfer from general information, such as color blobs in images (early layers), to specific information related to a particular class (last layers), and thus the distilled images PFs representations tend to encode all this information through KIP training. In ScatterNet features case, the intuition for the visual interpretability of the distilled samples is that the feature representations conserve the energy of the original signal (or information within the image) as described in Oyallon & Mallat (2015). Therefore, the learned distilled images tend to learn more detailed patterns from the original images.

In the privacy realm, we evaluated DP-KIP performance in Table 2 using fully connected infinite-width NTK (DP-KIP FC-NTK) as private baseline and compared it to ScatterNet features (DP-KIP-ScatterNet). We consider $\epsilon \in \{1, 10\}$ and $\delta = 10^{-5}$. As in the non-private setting, the general trend is that creating more images per class improves the classification accuracy, while the gradient dimension in DP-SGD increases. Furthermore, the results show that DP-KIP-ScatterNet significantly improves the performance compared to DP-KIP using FC-NTK. In Fig. 2, we show the learned distilled images at $\epsilon = 1$ and $\epsilon = 10$. It is surprising that the images created at $\epsilon = 10$ are not humanly discernible, while the classifiers can still achieve a good classification performance. Detailed hyperparameter settings can be found in Sec. D.1 in Appendix.

In Table 3, we explore the performance of DP-KIP compared to other methods for private data generation (DP-CGAN, G-PATE, DataLens, GS-WGAN, DP-MERF, DP-Sinkhorn) and private gradient matching by Chen et al. (2022). We report the test accuracy of a ConvNet downstream classifier consisting of 3 blocks, where each block contains one Conv layer with 128 filters, Instance Normalization, ReLU activation and AvgPooling modules and a FC layer as the final output. The classifier is trained using private data and then, it is tested on real test data over 3 independent runs for each method. Here, the methods based on DP-KIP algorithm outperform existing methods at the same privacy level, achieving the best accuracy by using ScatterNet features (DP-KIP-ScatterNet).

Table 4 shows the KRR test accuracy for DP-KIP-ScatterNet with pixel corruption on CIFAR-10 for generating 1, 10 and 50 distilled images per class at different ϵ values. Here, we randomly fix some pixels from the distilled images to a noisy value and perform DP-KIP-ScatterNet on the rest of the pixels. As a general trend, we observe that KRR accuracy improves over all-pixel optimization (Table 2) at small proportions of corrupted pixels (1%, 5%). We also observe that increasing the number of distilled samples (10 and 50 images/class) enables an increase in the proportion of corrupted pixels up to 10% compared to all-pixel optimization KRR accuracy.

Table 1: KRR test accuracy on Image datasets for KIP. The average over five independent runs. KIP e-NTK, KIP ScatterNet and KIP PFs outperform KIP FC-NTK. KIP ScatterNet outperforms or performs similar to KIP ConvNet-NTK. Best accuracy is marked in bold and overlapping overlapping best accuracies are highlighted in italics.

	Imgs/ Class	KIP FC-NTK	KIP ConvNet-NTK		KIP e-NTK	KIP ScatterNet	KIP PFs
			no aug	aug		no aug	no aug
MNIST	1	89.3 ± 0.1	97.3 ± 0.1	96.5 ± 0.1	93.6 ± 0.4	**98.2 ± 0.1**	94.6 ± 0.6
	10	96.6 ± 0.1	***99.1 ± 0.1***	***99.1 ± 0.1***	96.7 ± 0.2	*99.0 ± 0.1*	97.7 ± 0.1
	50	97.6 ± 0.1	*99.4 ± 0.1*	***99.5 ± 0.1***	97.6 ± 0.1	*99.4 ± 0.1*	97.9 ± 0.1
FASHION	1	80.3 ± 0.4	82.9 ± 0.2	76.7 ± 0.2	75.2 ± 0.3	**87.2 ± 0.2**	84.6 ± 0.2
	10	84.8 ± 0.4	**91.0 ± 0.1**	88.8 ± 0.1	76.2 ± 0.2	89.5 ± 0.1	88.1 ± 0.4
	50	86.1 ± 0.1	**92.4 ± 0.1**	91.0 ± 0.1	79.5 ± 0.1	90.6 ± 0.1	89.3 ± 0.3
SVHN	1	25.4 ± 0.3	62.4 ± 0.2	64.3 ± 0.4	19.9 ± 0.3	**77.4 ± 0.2**	62.1 ± 0.2
	10	59.7 ± 0.5	79.3 ± 0.1	81.1 ± 0.5	21.1 ± 0.4	**84.4 ± 0.1**	81.2 ± 0.2
	50	69.7 ± 0.1	82.0 ± 0.1	84.3 ± 0.1	27.5 ± 0.2	**86.4 ± 0.1**	82.6 ± 0.2
CIFAR-10	1	39.3 ± 1.6	**64.7 ± 0.2**	63.4 ± 0.1	32.0 ± 0.7	60.2 ± 0.1	44.7 ± 0.6
	10	49.1 ± 1.1	**75.6 ± 0.2**	75.5 ± 0.1	40.0 ± 0.3	66.2 ± 0.2	53.3 ± 0.5
	50	52.1 ± 0.8	78.2 ± 0.2	**80.6 ± 0.1**	42.4 ± 0.2	68.5 ± 0.1	54.4 ± 0.3
CIFAR-100	1	14.5 ± 0.4	**34.9 ± 0.1**	33.3 ± 0.3	9.4 ± 0.2	27.4 ± 0.2	19.6 ± 0.3
	10	12.2 ± 0.2	47.9 ± 0.2	**49.5 ± 0.3**	12.5 ± 0.2	35.8 ± 0.1	23.5 ± 0.3
	50	12.3 ± 0.2	-	-	13.7 ± 0.2	45.7 ± 0.1	24.9 ± 0.4

Table 2: KRR test accuracy on Image datasets for DP-KIP FC-NTK and DP-KIP-ScatterNet at $\epsilon = 1, 10$ and $\delta = 10^{-5}$. The average over five independent runs. DP-KIP-ScatterNet outperforms DP-KIP FC-NTK for all image datasets.

	Imgs/ Class	DP-KIP FC-NTK		DP-KIP-ScatterNet	
		$\epsilon = 1$	$\epsilon = 10$	$\epsilon = 1$	$\epsilon = 10$
MNIST	1	82.7 ± 0.1	85.2 ± 0.1	**94.7 ± 0.3**	**96.1 ± 0.2**
	10	87.5 ± 0.4	89.3 ± 0.3	**96.3 ± 0.1**	**97.4 ± 0.1**
	50	92.7 ± 0.2	93.4 ± 0.1	**97.2 ± 0.3**	**98.1 ± 0.1**
FASHION	1	76.9 ± 0.1	78.3 ± 0.1	**77.5 ± 0.1**	**82.1 ± 0.2**
	10	77.7 ± 0.1	78.7 ± 0.4	**83.3 ± 0.2**	**86.2 ± 0.1**
	50	78.8 ± 0.1	81.1 ± 0.1	**84.7 ± 0.3**	**87.6 ± 0.1**
SVHN	1	24.9 ± 0.3	25.2 ± 0.2	**55.3 ± 0.6**	**68.6 ± 0.2**
	10	40.5 ± 1.2	47.2 ± 0.6	**66.4 ± 0.3**	**74.4 ± 0.2**
	50	52.7 ± 0.4	56.6 ± 0.4	**71.4 ± 0.4**	**76.7 ± 0.2**
CIFAR-10	1	36.7 ± 0.3	37.3 ± 0.1	**46.6 ± 0.5**	**50.4 ± 0.4**
	10	38.3 ± 0.3	39.7 ± 0.3	**48.6 ± 0.4**	**54.4 ± 0.3**
	50	40.8 ± 0.2	43.7 ± 0.1	**49.7 ± 0.5**	**58.7 ± 0.2**
CIFAR-100	1	9.9 ± 0.6	11.1 ± 0.1	**12.7 ± 0.2**	**17.5 ± 0.2**
	10	10.1 ± 0.3	12.1 ± 0.4	**14.1 ± 0.1**	**21.1 ± 0.1**
	50	11.3 ± 0.3	13.6 ± 0.2	**16.2 ± 0.1**	**25.2 ± 0.1**

5.2 Tabular data

In the following we present DP-KIP results applied to eight different tabular datasets for imbalanced data. These datasets contain both numerical and categorical input features and are described in detail in Sec. C in Appendix. To evaluate the utility of the distilled samples, we train 12 commonly used classifiers on the distilled data samples and then evaluate their performance on real data for 5 independent runs. In tabular data experiments, we tested our method only with infinite-width fully-connected NTKs. The motivation behind this is that by construction, ScatterNet features are designed to detect and describe local image features and thus, may not be a suitable option for tabular datasets.

For datasets with binary labels, we use the area under the receiver characteristics curve (ROC) and the area under the precision recall curve (PRC) as evaluation metrics, and for multi-class datasets, we use F1 score. Table 5 shows the average over the classifiers (averaged again over the 5 independent runs) trained on the synthetic privated generated samples for DP-CGAN Torkzadehmahani et al. (2019), DP-GAN (Xie et al., 2018), DP-MERF (Harder et al., 2021), DP-HP (Vinaroz et al., 2022) and DP-NTK (Yang et al., 2023) and

Table 3: **Left:** Test accuracy of ConvNet downstream classifier trained on synthetic/distilled data with $\delta = 10^{-5}$. **Right:** Test accuracy of ConvNet downstream classifier trained with fixed $\epsilon = 10$ and $\delta = 10^{-5}$ and varying the number of distilled samples per class. In concurrent work Zheng & Li (2023), the test accuracy on the same classifier tested on 50 MNIST distilled samples created at $\epsilon = 6.12$ and $\delta = 10^{-5}$ is 97.35%; that on 50 Fashion-MNIST distilled samples created at $\epsilon = 5.45$ and $\delta = 10^{-5}$ is 82.72%.

	MNIST		FASHION	
	$\epsilon = 1$	$\epsilon = 10$	$\epsilon = 1$	$\epsilon = 10$
DP-CGAN	38.1	52.5	26.3	50.2
G-PATE	58.8	80.9	58.1	69.3
DataLens	71.2	80.7	64.8	70.6
GS-WGAN	48.3	84.9	39.2	63.1
DP-MERF	72.7	85.7	61.2	72.4
DP-Sinkhorn	70.2	83.2	56.3	71.1
DP-KIP FC-NTK (10 Imgs/Class)	93.53	95.39	87.7	89.1
DP-KIP-ScatterNet (10 Imgs/Class)	**94.5**	**98.0**	**88.0**	**89.3**
Chen et al. (2022) (20 Imgs/Class)	80.9	95.6	70.2	77.7
DP-KIP FC-NTK (20 Imgs/Class)	**97.78**	97.96	88.3	90.2
DP-KIP-ScatterNet (20 Imgs/Class)	96.3	**98.4**	**88.5**	**90.9**

	MNIST			FASHION		
Imgs/Class	10	20	full	10	20	full
Real	93.6	95.9	99.6	74.4	77.4	93.5
DP-SGD	-	-	96.5	-	-	82.9
DP-CGAN	57.4	57.1	52.5	51.4	53.0	50.2
G-PATE	70.7	73.6	80.9	58.6	62.4	69.3
DataLens	56.5	66.3	80.7	61.1	62.7	70.6
GS-WGAN	83.3	85.5	84.9	58.7	59.5	63.1
DP-MERF	80.2	83.2	85.7	66.6	67.9	72.4
DP-Sinkhorn	74.8	80.5	83.2	67.4	68.5	71.1
Chen et al. (2022)	94.9	95.6	-	75.6	77.7	-
DP-KIP FC-NTK	95.39	97.96	-	89.1	90.2	-
DP-KIP-ScatterNet	**98.0**	**98.4**	-	**89.3**	**90.9**	-

Table 4: KRR test accuracy vs. % of corrupted pixels on CIFAR-10. Best accuracy results are obtained for DP-KIP-ScatterNet at different epsilon values and $\delta = 10^{-5}$ with 1% and 5% of corrupted pixels.

Imgs/Class	Corrupted pixels (%)	$\epsilon = 0.1$	$\epsilon = 1$	$\epsilon = 10$	$\epsilon = 100$
1	0	-	46.6 ± 0.5	50.4 ± 0.4	-
	1	**46.8 ± 0.2**	50.4 ± 0.3	**52.6 ± 0.4**	**54.5 ± 0.3**
	5	45.7 ± 0.3	**50.9 ± 0.4**	51.1 ± 0.5	53.1 ± 0.3
	10	43.2 ± 0.2	45.9 ± 0.4	49.9 ± 0.3	52.4 ± 0.6
	40	41.4 ± 0.3	45.2 ± 0.2	48.3 ± 0.2	51.0 ± 0.1
	80	38.4 ± 0.3	40.1 ± 0.3	45.9 ± 0.4	49.5 ± 0.3
10	0	-	48.6 ± 0.4	54.4 ± 0.3	-
	1	52.5 ± 0.3	**54.9 ± 0.1**	**55.7 ± 0.2**	**56.2 ± 0.1**
	5	**53.9 ± 0.1**	54.1 ± 0.1	54.6 ± 0.1	55.3 ± 0.1
	10	46.3 ± 0.2	52.1 ± 0.1	53.6 ± 0.2	53.9 ± 0.1
	40	46.2 ± 0.3	47.5 ± 0.2	52.2 ± 0.1	52.7 ± 0.2
	80	41.6 ± 0.1	46.5 ± 0.2	50.2 ± 0.1	50.9 ± 0.1
50	0	-	49.76 ± 0.5	58.7 ± 0.2	-
	1	55.2 ± 0.1	**56.4 ± 0.1**	**59.3 ± 0.1**	**59.8 ± 0.2**
	5	**55.4 ± 0.1**	56.3 ± 0.1	59.1 ± 0.1	59.3 ± 0.1
	10	54.1 ± 0.1	55.2 ± 0.2	58.9 ± 0.1	59.2 ± 0.1
	40	47.1 ± 0.1	48.8 ± 0.1	55.7 ± 0.1	56.4 ± 0.1
	80	46.5 ± 0.2	47.3 ± 0.1	53.3 ± 0.1	55.2 ± 0.1

trained on the privately distilled samples for DP-KIP FC-NTK under the same privacy budget $\epsilon = 1$ and $\delta = 10^{-5}$. Details on hyperparameter settings and classifiers used in evaluation can be found in Sec. D.2 in Appendix.

For private synthetic methods we generate as many samples as the target dataset contains while for DP-KIP FC-NTK we set the images per class to 10. Unsurprisingly, our method outperforms the general data generation methods at the same privacy level.

6 Summary and Discussion

In this paper, we propose alternative kernels to infinite-dimensional NTK in KIP algorithm. The alternative kernels reduce computational resources required to run the algorithm (down to a single GPU), while maintaining (even improving in particular cases) the quality of the distilled dataset. The computational requirement reduction has a positive impact on the carbon footprint and makes the algorithm accessible for a wider range of practitioners. Among all kernels tested, we empirically show that the kernel defined by ScatterNet features is the most suitable in image data experiments.

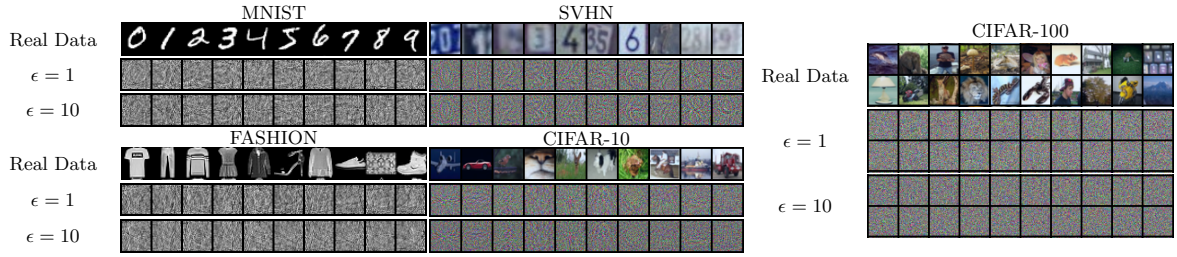

Figure 2: Generated image samples from DP-KIP-ScatterNet for different ϵ values.

Table 5: Performance comparison on Tabular datasets at $\epsilon = 1$ and $\delta = 10^{-5}$. The average over five independent runs. For DP-KIP FC-NTK, 10 images per class are distilled and for private synthetic methods, as many samples as the original dataset contains are generated.

	Real		DP-CGAN $(1, 10^{-5})$-DP		DP-GAN $(1, 10^{-5})$-DP		DP-MERF $(1, 10^{-5})$-DP		DP-HP $(1, 10^{-5})$-DP		DP-NTK $(1, 10^{-5})$-DP		**DP-KIP FC-NTK** $(1, 10^{-5})$-DP	
	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC
adult	0.786	0.683	0.509	0.444	0.511	0.445	0.642	0.524	0.688	**0.632**	**0.695**	0.557	0.662	0.365
census	0.776	0.433	0.655	0.216	0.529	0.166	0.685	0.236	0.699	0.328	0.71	**0.424**	**0.766**	0.408
cervical	0.959	0.858	0.519	0.200	0.485	0.183	0.531	0.176	0.616	0.312	**0.631**	**0.335**	0.622	0.316
credit	0.924	0.864	0.664	0.356	0.435	0.150	0.751	0.622	0.786	0.744	0.821	**0.759**	**0.892**	0.610
epileptic	0.808	0.636	0.578	0.241	0.505	0.196	0.605	0.316	0.609	0.554	0.648	0.326	**0.654**	**0.585**
isolet	0.895	0.741	0.511	0.198	0.540	0.205	0.557	0.228	0.572	0.498	0.53	0.205	**0.615**	**0.955**
	F1		F1		F1		F1		F1		F1		F1	
covtype	0.820		0.285		0.492		0.467		0.537		0.552		**0.582**	
intrusion	0.971		0.302		0.251		**0.892**		0.890		0.717		0.873	

As illustrated by Carlini et al. (2022), data distillation algorithms lack inherent privacy guarantees. In response to this privacy concern, we develop DP-KIP algorithm using DP-SGD on the existing KIP algorithm and implement it in JAX. Experimental results show that our proposed method outperforms existing data distillation methods at the same privacy level under image classification tasks with DP-KIP-ScatterNet achieving the highest performance. Additionally, we conduct experiments involving corrupted pixels on DP-KIP-ScatterNet on CIFAR10. The experiments with corrupted pixels demonstrate that fixing a small percentage of randomly selected pixels (1%, 5%) and applying DP-KIP-ScatterNet on the remaining pixels positively impacts the quality of the resulting distilled samples, surpassing the all-pixel optimization in terms of KRR accuracy.

We also evaluated DP-KIP in tasks involving the classification of tabular data using the infinitely-wide fully-connected NTK. The experimental findings indicate that, despite distilling only 10 images per class, our approach typically outperforms differentially private methods for data generation, which generate samples matching the size of the target dataset. To the best of our knowledge, this is the first work that implements a data distillation algorithm to a tabular data classification problem.

In future work, we aim to enhance our proposed method further, narrowing the accuracy gap with respect to KIP performance, especially in the context of image datasets.

Acknowledgments

We thank our anonymous reviewers for their constructive feedback, which has helped significantly improve our manuscript. We thank the Digital Research Alliance of Canada (Compute Canada) for its computational resources and services. M. Vinaroz was funded by the Canada CIFAR AI Chairs program (at AMII), as a visiting international research student (VIRS) in the Department of Computer Science at the University of British Columbia. M. Park was partially funded by Novo Nordisk Fonden RECUIT grant no.0065800.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, oct 2016. doi: 10.1145/2976749.2978318. URL https://doi.org/10.1145%2F2976749.2978318.

N. Aronszajn. Theory of reproducing kernels. *Trans Am Math Soc*, 68(3):337–404, 1950. ISSN 00029947. URL www.jstor.org/stable/1990404.

Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2019. URL https://openreview.net/pdf?id=rkl4aESeUH.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with sinkhorn divergence. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12480–12492. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/67ed94744426295f96268f4ac1881b46-Paper.pdf.

Nicholas Carlini, Vitaly Feldman, and Milad Nasr. No free lunch in "privacy for free: How does dataset condensation help privacy", 2022. URL https://arxiv.org/abs/2209.14987.

Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. GS-WGAN: A gradient-sanitized approach for learning differentially private generators. In *Advances in Neural Information Processing Systems 33*, 2020.

Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Private set generation with discriminative information. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=mxnxRw8jiru.

Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5378–5396. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/dong22c.html.

Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.*, 9:211–407, aug 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL http://dx.doi.org/10.1561/0400000042.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Eurocrypt*, volume 4004, pp. 486–503. Springer, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin (eds.), *Theory of Cryptography*, pp. 265–284, Berlin, Heidelberg, 2006b. Springer Berlin Heidelberg. ISBN 978-3-540-32732-5.

Sam Fletcher and Md Zahidul Islam. A differentially private random decision forest using reliable signal-to-noise ratios. In Bernhard Pfahringer and Jochen Renz (eds.), *AI 2015: Advances in Artificial Intelligence*, pp. 192–203. Springer International Publishing, 2015. ISBN 978-3-319-26350-2.

Frederik Harder, Kamil Adamczewski, and Mijung Park. DP-MERF: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*,

volume 130 of *Proceedings of Machine Learning Research*, pp. 1819–1827. PMLR, 13–15 Apr 2021. URL http://proceedings.mlr.press/v130/harder21a.html.

Frederik Harder, Milad Jalali, Danica J. Sutherland, and Mijung Park. Pre-trained perceptual features improve differentially private image generation. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=R6W7zkMz0P.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. What makes imagenet good for transfer learning?, 2016.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 694–711, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46475-6.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/0d1a9651497a38d8b1c3871c84528bd4-Paper.pdf.

Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15156–15172. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/ad086f59924fffe0773f8d0ca22ea712-Paper.pdf.

Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 12245–12254, 2020.

Yunhui Long, Boxin Wang, Zhuolin Yang, Bhavya Kailkhura, Aston Zhang, Carl Gunter, and Bo Li. G-PATE: Scalable differentially private data generator via private aggregation of teacher discriminators. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 2965–2977. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/171ae1bbb81475eb96287dd78565b38b-Paper.pdf.

Noel Loo, Ramin Hasani, Mathias Lechner, and Daniela Rus. Dataset distillation fixes dataset reconstruction attacks. *arXiv preprint arXiv:2302.01428*, 2023.

Wojciech Masarczyk and Ivona Tautkute. Reducing catastrophic forgetting with learning on synthetic data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 252–253, 2020.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=l-PrrQrK0QR.

Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 5186–5198. Curran Associates, Inc., 2021b. URL https://proceedings.neurips.cc/paper/2021/file/299a23a2291e2126b91d54f3601ec162-Paper.pdf.

E. Oyallon and S. Mallat. Deep roto-translation scattering for object classification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2865–2873, Los Alamitos, CA, USA, jun 2015. IEEE Computer Society. doi: 10.1109/CVPR.2015.7298904. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298904.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Andrea Rosasco, Antonio Carta, Andrea Cossu, Vincenzo Lomonaco, and Davide Bacciu. Distilled replay: Overcoming forgetting through synthetic samples. In *International Workshop on Continual Semi-Supervised Learning*, pp. 104–117. Springer, 2021.

Mattia Sangermano, Antonio Carta, Andrea Cossu, and Davide Bacciu. Sample condensation in online continual learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 01–08. IEEE, 2022.

Cicero Nogueira Dos Santos, Youssef Mroueh, Inkit Padhi, and Pierre Dognin. Learning implicit generative models by matching perceptual features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4460–4469, 2019. doi: 10.1109/ICCV.2019.00456.

Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *International Conference on Learning Representations*, 2017.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. DP-CGAN: Differentially private synthetic data and label generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

Florian Tramer and Dan Boneh. Differentially private learning needs better features (or much more data). In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YTWGvpFOQD-.

Margarita Vinaroz, Mohammad-Amin Charusaie, Frederik Harder, Kamil Adamczewski, and Mi Jung Park. Hermite polynomial features for private data generation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22300–22324. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/vinaroz22a.html.

Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. Datalens: Scalable privacy preserving training via gradient compression and aggregation. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, pp. 2146–2168, New York, NY, USA,

2021. Association for Computing Machinery. ISBN 9781450384544. doi: 10.1145/3460120.3484579. URL https://doi.org/10.1145/3460120.3484579.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. *CoRR*, abs/1811.10959, 2018. URL http://arxiv.org/abs/1811.10959.

Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. volume 89 of *Proceedings of Machine Learning Research*, pp. 1226–1235. PMLR, April 2019.

Felix Wiewel and Bin Yang. Condensed composite memory continual learning. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. In *International Conference on Machine Learning*, pp. 5393–5402. PMLR, 2018.

Lechao Xiao, Jeffrey Pennington, and Samuel S. Schoenholz. Disentangling trainability and generalization in deep neural networks, 2020.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

Yilin Yang, Kamil Adamczewski, Danica J. Sutherland, Xiaoxiao Li, and Mijung Park. Differentially private neural tangent kernels for privacy-preserving data generation, 2023.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014a. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/375c71349b295fbe2dcdca9206f20a06-Paper.pdf.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014b.

Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning*, pp. 12674–12685. PMLR, 2021.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6514–6523, 2023.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=mSAKhLYLSsl.

Tianhang Zheng and Baochun Li. Differentially private dataset condensation, 2023. URL https://openreview.net/forum?id=H8XpqEkbua_.

Appendix

A Clipping effect of the gradients

In this section we empirically show the effect of clipping the gradients for DP-KIP-ScatterNet in terms of KRR accuracy. Table 6 shows the accuracy for 1 Imgs/Class for MNIST at $\epsilon = 1, 10$ and $\delta = 10^{-5}$.

Table 6: KRR accuracy for 1 Imgs/Class distilled dataset on MNIST at different C.

	C					
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
$\epsilon = 1$	81.5 ± 0.4	**94.7 ± 0.3**	93.7 ± 0.4	93.9 ± 0.4	93.5 ± 0.5	90.5 ± 0.2
$\epsilon = 10$	80.2 ± 0.6	**96.1 ± 0.2**	94.8 ± 0.2	94.9 ± 0.1	94.7 ± 0.2	94.2 ± 0.4

B Signal-to-Noise ratio of privatizing the feature maps

Definition B.1. *Signal-to-Noise Ratio (SNR) Fletcher & Islam (2015) The Signal-to-Noise Ratio of a measurement can be expressed as* $\frac{signal}{noise}$*, where the signal and noise are expressed in the same units. An alternative way of writing this is:*

$$SNR = \frac{\mu}{\sigma} \tag{7}$$

where μ *is the signal mean or expected value and* σ *is the standard deviation of the noise.*

Now, we want to to see what's the SNR in DP-KIP scenario for privatizing data dependent feature maps once and reuse them during training. Consider a target dataset $\mathcal{D}_t = \{(\mathbf{x}_{t_i}, y_{t_i})\}_{i=1}^n$ with input features $\mathbf{x}_{t_i} \in \mathbb{R}^D$ and scalar labels y_{t_i}. Let $k\colon \mathbb{R}^D \times \mathbb{R}^D$ be a positive definite kernel. By Moore–Aronszajn theorem Aronszajn (1950), here exists a unique reproducing kernel Hilbert space of functions on $\mathbb{R}^D$ for which k is a reproducing kernel and thus, we can find a feature map, $\phi\colon \mathbb{R}^D \to \mathcal{H}$ such that $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$ for any pair of datapoints $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$.

We start by deriving the sensitivity of the feature map for neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$. Without loss of generality, we consider that $\mathcal{D}$ and $\mathcal{D}'$ differ in the last datapoint such that $\mathbf{x}_{t_n} \neq \mathbf{x}'_{t_n}$ and normalized feature maps, $\|\phi(\mathbf{x}_{t_i})\| = 1, \forall i \in [n]$.

$$\Delta_\phi^2 = \max_{\mathcal{D},\mathcal{D}'} \left\| \begin{bmatrix} \phi(\mathbf{x}_{t_1}) \\ \phi(\mathbf{x}_{t_2}) \\ \vdots \\ \phi(\mathbf{x}_{t_n}) \end{bmatrix} - \begin{bmatrix} \phi(\mathbf{x}_{t_1}) \\ \phi(\mathbf{x}_{t_2}) \\ \vdots \\ \phi(\mathbf{x}'_{t_n}) \end{bmatrix} \right\|_2 = \max_{\mathcal{D},\mathcal{D}'} \left\| \phi(\mathbf{x}_{t_n}) - \phi(\mathbf{x}'_{t_n}) \right\|_2 \leq 2 \tag{8}$$

where the last inequality is due to the triangular inequality.

Now, we compute the SNR for the the feature map representations of datapoints in $\mathcal{D}_t$:

$$SNR_{\phi(\mathbf{x}_t)} = \frac{\left\| \begin{bmatrix} \phi(\mathbf{x}_{t_1}) \\ \phi(\mathbf{x}_{t_2}) \\ \vdots \\ \phi(\mathbf{x}_{t_n}) \end{bmatrix} \right\|_F}{\sqrt{n\Delta_\phi^2\sigma^2}} = \frac{\sqrt{\|\phi(\mathbf{x}_{t_1})\|_2^2 + \cdots + \|\phi(\mathbf{x}_{t_n})\|_2^2}}{\sqrt{n\Delta_\phi^2\sigma^2}} = \frac{\sqrt{n}}{2\sqrt{n}\sigma} = \frac{1}{2\sigma} \tag{9}$$

Therefore, we observe that signal-to-noise ratio of the feature maps is inversevely proportional to the standard deviation parameter σ. This implies that the noise will dominate the signal-to-noise ratio unless σ is a relatively small value which will only be the case for large ϵ values since $\sigma = \frac{\alpha\Delta_\phi}{\sqrt{2\epsilon_{RDP}(\alpha)}}$ and $\epsilon(\delta) = \min_{\alpha>1} \frac{\log(1/\delta)}{\alpha-1} + \epsilon_{RDP}(\alpha-1)$ for a given $\delta \ll 1/n$.

C Tabular datasets description

Table 7 contains detail information about the 8 different tabular datasets used in Sec. 5.2.

Table 7: Tabular datasets. Size, number of classes and feature types descriptions.

dataset	# samps	# classes	# features
isolet	4366	2	617 num
covtype	406698	7	10 num, 44 cat
epileptic	11500	2	178 num
credit	284807	2	29 num
cervical	753	2	11 num, 24 cat
census	199523	2	7 num, 33 cat
adult	48842	2	6 num, 8 cat
intrusion	394021	5	8 cat, 6 ord, 26 num

D Expermental details

D.1 Image data

Here we provide hyperparameter settings used during image data KIP experiments. Table 8, Table 9 and Table 10 show the number of epochs, batch size, learning rate and regularization parameter λ used in KIP e-NTK, KIP PFs and KIP ScattterNet respectively.

Table 8: KIP e-NTK hyperparameter settings for Image data.

	Imgs/Class	epochs	batch size	learning rate	λ
MNIST	1	1000	1000	$5 \cdot 10^{-3}$	10^{-7}
	10	1000	1000	$5 \cdot 10^{-3}$	10^{-7}
	50	10	500	10^{-2}	10
FASHION	1	100	100	10^{-3}	10^{-2}
	10	100	100	10^{-3}	10^{-1}
	50	10	100	10^{-2}	10^{-1}
SVHN	1	100	2000	10^{-2}	10^{-3}
	10	100	2000	10^{-2}	10^{-4}
	50	10	1000	10^{-4}	10^{-3}
CIFAR-10	1	100	200	10^{-2}	10^{-3}
	10	100	1000	10^{-2}	10^{-4}
	50	100	2000	10^{-2}	10^{-2}
CIFAR-100	1	100	1000	10^{-1}	10^{-3}
	10	1000	50	10^{-1}	10^{-3}
	50	1000	50	10^{-1}	10^{-2}

Table 9: KIP PFs hyperparameter settings for Image data.

	Imgs/Class	epochs	batch size	learning rate	λ
MNIST	1	100	1000	10^{-2}	10^{-6}
	10	100	5000	10^{-2}	10^{-5}
	50	100	2000	10^{-2}	10^{-6}
FASHION	1	1000	2000	10^{-2}	10^{-5}
	10	1000	1000	10^{-2}	10^{-5}
	50	1000	5000	10^{-3}	10^{-5}
SVHN	1	100	2000	10^{-2}	10^{-3}
	10	100	2000	10^{-2}	10^{-3}
	50	1000	100	10^{-4}	10^{-2}
CIFAR-10	1	1000	2000	10^{-2}	10^{-4}
	10	1000	5000	10^{-3}	10^{-3}
	50	1000	2000	10^{-3}	10^{-2}
CIFAR-100	1	1000	1000	10^{-2}	10^{-3}
	10	1000	200	10^{-2}	10^{-1}
	50	100	200	10^{-2}	10^{-3}

Table 10: KIP ScatterNet hyperparameter settings for Image data.

	Imgs/Class	epochs	batch size	learning rate	λ
MNIST	1	1000	200	10^{-4}	1
	10	2000	100	10^{-4}	10^{-5}
	50	2000	5000	10^{-4}	10^{-5}
FASHION	1	1000	1000	10^{-3}	1
	10	2000	2000	10^{-3}	10^{-6}
	50	2000	2000	$5 \cdot 10^{-3}$	10^{-2}
SVHN	1	1000	200	10^{-4}	1
	10	1000	1000	10^{-3}	10^{-4}
	50	8000	5000	$5 \cdot 10^{-3}$	10^{-3}
CIFAR-10	1	1000	2000	10^{-3}	1
	10	1000	2000	10^{-3}	10^{-6}
	50	1000	5000	10^{-3}	10^{-3}
CIFAR-100	1	10000	2000	10^{-2}	10^{-3}
	10	1000	5000	10^{-2}	10^{-3}
	50	2000	1000	10^{-2}	10^{-1}

Here we provide the details of the DP-KIP training procedure we used on the image data experiments. Table 11 and Table 12 show the number of epochs, batch size, learning rate, clipping norm C and regularization parameter λ used during training for each image classification dataset at the corresponding images per class distilled for DP-KIP FC-NTK. Table 13 and Table 14 show the different hyperparameter settings used in DP-KIP-ScatterNet experiments.

D.2 Tabular data

Table 15 contains DP-KIP FC-NTK hyperparameter settings used in tabular data experiments. Table 16 describes the hyperparameter setting for training the downstream classifiers used in tabular data experiments.

Table 11: DP-KIP FC-NTK hyperparameter settings for Image data for $\epsilon = 1$ and $\delta = 10^{-5}$.

	Imgs/Class	epochs	batch size	learning rate	C	λ
MNIST	1	10	500	$5 \cdot 10^{-3}$	10^{-2}	10^{-6}
	10	10	500	10^{-2}	10^{-6}	10^{-6}
	50	10	500	10^{-2}	10^{-6}	10^{-7}
FASHION	1	10	500	$5 \cdot 10^{-2}$	10^{-6}	10^{-6}
	10	10	500	0.1	10^{-6}	10^{-5}
	50	10	200	10^{-2}	10^{-2}	10^{-6}
SVHN	1	10	50	10^{-1}	10^{-6}	10^{-6}
	10	10	500	$5 \cdot 10^{-2}$	10^{-6}	10^{-6}
	50	10	500	10^{-2}	10^{-5}	10^{-2}
CIFAR-10	1	20	200	10^{-2}	10^{-4}	10^{-5}
	10	10	500	$5 \cdot 10^{-2}$	10^{-5}	10^{-6}
	50	10	500	$5 \cdot 10^{-2}$	10^{-3}	10^{-6}
CIFAR-100	1	10	200	10^{-2}	10^{-5}	10^{-7}
	10	10	100	10^{-2}	10^{-4}	10^{-7}
	50	10	50	10^{-2}	10^{-3}	10^{-7}

Table 12: DP-KIP FC-NTK hyperparameter settings for Image data for $\epsilon = 10$ and $\delta = 10^{-5}$.

	Imgs/Class	epochs	batch size	learning rate	C	λ
MNIST	1	10	500	$5 \cdot 10^{-3}$	10^{-5}	10^{-6}
	10	10	500	$5 \cdot 10^{-3}$	10^{-2}	10^{-6}
	50	10	500	$5 \cdot 10^{-3}$	10^{-5}	10^{-6}
FASHION	1	10	500	10^{-2}	10^{-3}	10^{-6}
	10	10	500	10^{-2}	10^{-2}	10^{-5}
	50	10	500	10^{-2}	10^{-2}	10^{-7}
SVHN	1	10	50	$5 \cdot 10^{-2}$	10^{-6}	10^{-6}
	10	10	500	10^{-1}	10^{-6}	10^{-6}
	50	10	500	10^{-2}	10^{-2}	10^{-7}
CIFAR-10	1	10	500	10^{-1}	10^{-6}	10^{-6}
	10	10	500	$5 \cdot 10^{-2}$	10^{-5}	10^{-6}
	50	10	500	$5 \cdot 10^{-2}$	10^{-5}	10^{-6}
CIFAR-100	1	10	100	10^{-2}	10^{-5}	10^{-6}
	10	10	100	10^{-2}	10^{-4}	10^{-6}
	50	10	100	10^{-2}	10^{-3}	10^{-6}

Table 13: DP-KIP-ScatterNet hyperparameter settings for Image data for $\epsilon = 1$ and $\delta = 10^{-5}$.

	Imgs/Class	epochs	batch size	learning rate	C	λ
MNIST	1	20	5000	10^{-1}	10^{-2}	10^{-4}
	10	30	500	10^{-1}	10^{-7}	10^{-3}
	50	30	200	10^{-1}	10^{-2}	10^{-2}
FASHION	1	50	2000	10^{-1}	10^{-4}	10^{-7}
	10	40	1000	10^{-2}	10^{-4}	10^{-3}
	50	20	200	10^{-2}	10^{-5}	10^{-2}
SVHN	1	50	2000	10^{-1}	10^{-4}	10^{-7}
	10	40	1000	10^{-2}	10^{-4}	10^{-3}
	50	20	200	10^{-2}	10^{-5}	10^{-2}
CIFAR-10	1	40	2000	10^{-1}	10^{-6}	10^{-6}
	10	20	500	10^{-1}	10^{-6}	10^{-3}
	50	30	100	10^{-2}	10^{-2}	10^{-1}
CIFAR-100	1	30	200	10^{-4}	10^{-6}	10^{-2}
	10	10	50	10^{-4}	10^{-6}	10^{-1}
	50	10	50	10^{-4}	10^{-6}	10^{-2}

Table 14: DP-KIP-ScatterNet hyperparameter settings for Image data at $\epsilon = 10$ and $\delta = 10^{-5}$.

	Imgs/Class	epochs	batch size	learning rate	C	λ
MNIST	1	30	500	10^{-1}	10^{-2}	10^{-6}
	10	50	2000	10^{-1}	10^{-6}	10^{-4}
	50	30	200	10^{-1}	10^{-7}	10^{-1}
FASHION	1	50	1000	10^{-1}	10^{-2}	10^{-4}
	10	50	2000	10^{-2}	10^{-1}	10^{-3}
	50	50	200	10^{-2}	10^{-2}	10^{-3}
SVHN	1	50	2000	10^{-2}	10^{-2}	10^{-6}
	10	50	500	10^{-2}	10^{-5}	10^{-3}
	50	50	100	10^{-2}	10^{-2}	10^{-3}
CIFAR-10	1	50	1000	10^{-2}	10^{-2}	10^{-5}
	10	50	500	10^{-2}	10^{-4}	10^{-4}
	50	50	100	10^{-2}	10^{-5}	10^{-4}
CIFAR-100	1	50	1000	10^{-1}	10^{-6}	10^{-5}
	10	50	50	10^{-1}	10^{-6}	10^{-2}
	50	50	50	10^{-1}	10^{-6}	10^{-2}

Table 15: DP-KIP FC-NTK hyperparameter settings for tabular data.

dataset	epochs	batch size (%)	learning rate	C	λ	subsampling rate
isolet	10	0.8	10^{-3}	10^{-3}	10^{-6}	10^{-2}
covtype	10	0.02	10^{-1}	10^{-2}	10^{-7}	$7 \cdot 10^{-2}$
epileptic	10	0.07	10^{-2}	10^{-3}	10^{-6}	$3 \cdot 10^{-1}$
credit	10	0.07	10^{-2}	10^{-1}	10^{-6}	10^{-2}
cervical	10	0.6	10^{-4}	10^{-3}	10^{-6}	$2 \cdot 10^{-2}$
census	10	0.4	$5 \cdot 10^{-1}$	10^{-1}	10^{-6}	$4 \cdot 10^{-2}$
adult	10	0.8	10^{-2}	10^{-1}	10^{-6}	$7 \cdot 10^{-2}$
intrusion	10	0.05	10^{-4}	10^{-4}	10^{-6}	10^{-2}

Table 16: Hyperparameter settings for downstream classifiers used in tabular data experiments. Models are taken from scikit-learn 0.24.2 and xgboost 0.90 python packages and hyperparameters have been set to achieve reasonable accuracy while limiting execution time. Parameters not listed are kept as default values.

Model	Parameters
Logistic Regression	solver: lbfgs, max_iter: 5000, multi_class: auto
Gaussian Naive Bayes	-
Bernoulli Naive Bayes	binarize: 0.5
LinearSVC	max_iter: 10000, tol: 1e-8, loss: hinge
Decision Tree	class_weight: balanced
LDA	solver: eigen, n_components: 9, tol: 1e-8, shrinkage: 0.5
Adaboost	n_estimators: 1000, learning_rate: 0.7, algorithm: SAMME.R
Bagging	max_samples: 0.1, n_estimators: 20
Random Forest	n_estimators: 100, class_weight: balanced
Gradient Boosting	subsample: 0.1, n_estimators: 50
MLP	-
XGB	colsample_bytree: 0.1, objective: multi:softprob, n_estimators: 50