"Private Prediction Strikes Back!" Private Kernelized Nearest Neighbors with Individual Rényi Filter

Yuqing Zhu[1] **Xuandong Zhao**[1] **Chuan Guo**[2] **Yu-Xiang Wang**[1]

[1]UC Santa Barbara
[2]FAIR / Meta AI

Abstract

Most existing approaches of differentially private (DP) machine learning focus on *private training*. Despite its many advantages, *private training* lacks the flexibility in adapting to incremental changes to the training dataset such as deletion requests from exercising GDPR's *right to be forgotten*. We revisit a long-forgotten alternative, known as *private prediction* [Dwork and Feldman, 2018], and propose a new algorithm named *Individual Kernelized Nearest Neighbor* (Ind-KNN). Ind-KNN is easily updatable over dataset changes and it allows precise control of the Rényi DP at an individual user level — a user's privacy loss is measured by the exact amount of her contribution to predictions; and a user is removed if her prescribed privacy budget runs out. Our results show that Ind-KNN consistently improves the accuracy over existing private prediction methods for a wide range of ϵ on four vision and language tasks. We also illustrate several cases under which Ind-KNN is preferable over private training with NoisySGD. Code is available at `https://github.com/jeremy43/Ind_kNN`.

1 INTRODUCTION

Differential privacy (DP; Dwork et al. [2006, 2014]) is a promising approach for mitigating privacy risks in machine learning (ML). The predominant setting for private ML is to produce the model learned from sensitive data using DP primitives, a.k.a. private training [Chaudhuri et al., 2011, Kasiviswanathan et al., 2011, Abadi et al., 2016]. The resulting trained model can then be safely deployed with peace of mind, because DP ensures that no individual training sample can be identified from the model itself or its downstream predictions.

Unfortunately, private training comes with several irky properties that hamper its real-life deployment. To begin, private training comes at a significant computation cost that can be restrictive in many applications. The NoisySGD algorithm [Abadi et al., 2016] requires per-sample gradient computation, which is much more computation- and memory-intensive than standard training.

Secondly, private training outputs a static model that cannot easily adapt to a changing dataset. For instance, additional data can arrive in a streaming fashion continuously. Also, training data could be mislabeled or corrupted [Chen et al., 2017, Jagielski et al., 2018] and the model needs to be patched accordingly. In addition, if the model is trained on user data, privacy regulations such as GDPR entitle the user to request the removal of their data from the model [Ginart et al., 2019, Guo et al., 2020, Bourtoule et al., 2021] with the so-called *right to be forgotten* [Mantelero, 2013]. These requirements can be satisfied by periodically re-training the model, but such an approach is not applicable to private training due to its high computation cost as well as privacy degradation after repeated training runs.

Thirdly, privacy training operates under a very strong threat model in which all downstream users can collude with each other in a coordinated attack on any individual training sample. Sometimes it makes sense to make realistic assumptions that limit the adversaries' information or resources. For example, Harvard's Privacy Tools project (now OpenDP) adopts a weaker threat model where each downstream user keeps the results to themselves [Gaboardi et al., 2016]. In this way, they each get to spend the privacy budget independently of everyone else and enjoy higher utility. Private training unfortunately does not have a means to benefit from having weaker adversaries.

To address these issues of private training, we revisit a viable but less-known alternative setting in differentially private machine learning known as *privacy-preserving prediction* (or simply *private prediction*) [Dwork and Feldman, 2018]. Instead of privately training the models and then using the model for predictions, private prediction aims at generating

Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023), PMLR 216:2586–2596.

Table 1: The amortized computational and privacy cost of answering $T = 2000$ queries on CIFAR-10. The median accuracy of all approaches across five independent runs is aligned to 96.0%. We estimate the amortized computational cost by calculating the averaged time spent (in seconds) to answer a single query, which is the total time of training divided by T in Linear NoisySGD [Feldman and Zrnic, 2021] and the total time of predictions divided by T in Private kNN [Zhu et al., 2020] and Ind-KNN. We use $\delta = 10^{-5}$. In the retraining scenarios, we assume that a retraining request is made every answering 100 queries, resulting in a total of 20 retraining requests among T queries.

	NoisySGD	NoisySGD (with retrain)	Private kNN	Ind-KNN (ours)	Ind-KNN+hashing (ours)
Computational cost (s)	0.008	0.16	0.12	0.25	0.04
Privacy loss (ϵ)	1.5	6.2	4.1	2.0	3.2

a sequence of predictions using the data directly. Notable methods include those that perturb the predictions of non-private models [Dwork and Feldman, 2018, Papernot et al., 2018, Bassily et al., 2018, Dagan and Feldman, 2020], or those that perturb the voting scores of the nearest neighbors [Zhu et al., 2020]. These methods require no changes to the (non-private) data workflow, and thus could more easily adapt to changing data.

From the privacy-utility trade-off point of view, the private prediction setting may appear to be counter-intuitive, because, for every prediction that it generates, a unit of privacy budget is spent. It is unreasonable to expect private prediction methods to outperform private training methods such as NoisySGD when we need to make many predictions. This was well-documented in the work of van der Maaten and Hannun [2020]. However, in the aforementioned situations when either frequent data updates are needed or a weaker adversary is assumed[1], private prediction methods can significantly outperform NoisySGD (See Table 1 and Figure 3 for an illustration). In fact, we will demonstrate that when combined with modern DP accounting techniques, data-adaptive DP algorithm design, and some clever reuse of previous predictions, a small privacy budget can answer thousands of queries without significantly increasing the privacy loss.

In this work, we propose *Individual Kernelized Nearest Neighbors* (Ind-KNN) — a new private prediction mechanism that significantly increases the number of queries one can answer with an individualized Rényi Differential Privacy accountant and other techniques. Intuitively, in KNN prediction, training samples that do not belong to the query's neighbor set do not contribute to the prediction, and hence their privacy cost should be negligible. We show that by slightly modifying KNN and leveraging Rényi filter [Feldman and Zrnic, 2021] to account for the privacy cost of each sample individually, we can realize this intuition in the privacy accounting and allow each training sample to participate in the query response until its own privacy budget is exhausted. In effect, common queries can be answered with relatively low privacy costs due to a large number of similar samples present in the training set.

[1]Consider the example of a recommendation system, each user makes a much smaller number of predictions than all users collectively.

Experimental results. We summarize our experimental results as follows:

1. We show that Ind-KNN consistently outperforms the private prediction benchmark, Private-kNN [Zhu et al., 2020], across four vision and language tasks for a range of epsilon between $[0.5, 2.0]$.
2. We demonstrate that Ind-KNN is a viable alternative to private training methods even in a static data setting. Our results indicate that Ind-KNN achieves higher accuracy than NoisySGD when answering less than 2000 queries on CIFAR-10 under $(1.0, 10^{-5})$-DP.
3. For frequent data updates, Ind-KNN significantly outperforms the private training benchmark Linear NoisySGD [Feldman and Zrnic, 2021]. As shown in Table 1, Linear NoisySGD requires a DP budget of $\epsilon = 6.2$ to achieve an accuracy of 96.0% on 2000 queries of CIFAR-10, while Ind-KNN only requires $\epsilon = 2.0$.
4. We describe two simple techniques that significantly enhance the computational efficiency and utility of Ind-KNN. First, we show that incorporating hashing tricks into Ind-KNN can provide a $6\times$ speedup in making predictions, with only a negligible drop in accuracy. Additionally, we propose to reuse the results of previous queries via post-processing, which allows Ind-KNN to answer an additional 1000 queries on CIFAR-10 without compromising in privacy or utility.

Related work and novelty. The problem of private prediction was pioneered by Dwork and Feldman [2018] as a weakened goal for private machine learning. Model-based approaches for private predictions either require analyzing the stability of model training [Dwork and Feldman, 2018, Dagan and Feldman, 2020] or to enforce stability of prediction via subsample-and-aggregate [Papernot et al., 2018, Bassily et al., 2018]. Our method is closest to Private kNN [Zhu et al., 2020] but uses kernel-weighted neighbors with a variable K instead of a fixed K. This change is critical for adapting the individual Rényi DP accountant (and filter) for our purpose. Other components such as adaptive noise-level, prediction reuse, and the fast hashing trick are new to

this paper. Technically, we apply the same individual Rényi filter [Feldman and Zrnic, 2021] that retires data samples when their privacy budget runs out. The difference is that we applied it to KNN rather than noisy gradient descent. KNN naturally has bounded support thus it is efficient to maintain the individual RDP accountants.

2 PRELIMINARIES

We start with the definition of differential privacy.

Definition 2.1 (Differential Privacy [Dwork et al., 2006])**.** A randomized algorithm $\mathcal{A} : \mathcal{X} \to \Theta$ is (ϵ, δ)-DP (differentially private) if for every pair of neighboring datasets $S, S' \in \mathcal{S}$, and every possible (measurable) output set $E \subseteq \Theta$ the following inequality holds:

$$\Pr[\mathcal{A}(S) \in E] \leq e^{\epsilon} \Pr[\mathcal{A}(S') \in E] + \delta.$$

Definition 2.2 (Rényi Differential Privacy [Mironov, 2017])**.** We say that a mechanism $\mathcal{A}$ is $(\alpha, \epsilon(\alpha))$-RDP with order $\alpha \in (1, \infty)$ if for all neighboring datasets S, S':

$$D_\alpha(\mathcal{A}(S) \| \mathcal{A}(S')) = \frac{1}{\alpha - 1} \log E_{\theta \sim \mathcal{A}(S')} \left[\left(\frac{p_{\mathcal{A}(S)}(\theta)}{p_{\mathcal{A}(S')}(\theta)} \right)^{\alpha} \right] \leq \epsilon(\alpha).$$

As $\alpha \to \infty$, RDP converges to the standard $(\epsilon, 0)$-DP. More generally, we can convert RDP to standard (ϵ, δ)-DP for any $\delta > 0$ using conversions from [Balle et al., 2020].

Privacy composition. RDP features a natural composition theorem that significantly simplifies privacy analysis over compositions, and often leads to a tighter privacy guarantee. If $\mathcal{A}_1(\cdot)$ is $(\alpha, \epsilon_{\mathcal{A}_1}(\alpha))$-RDP and $\mathcal{A}_2(\cdot)$ is $(\alpha, \epsilon_{\mathcal{A}_2}(\alpha))$-RDP, then the adaptive composition theorem for RDP says that $\epsilon_{\mathcal{A}_1 \circ \mathcal{A}_2}(\cdot)$ satisfies $(\alpha, \epsilon_{\mathcal{A}_1}(\alpha) + \epsilon_{\mathcal{A}_2}(\alpha))$-RDP.

Privacy-preserving prediction. We now formally state the setting of privacy-preserving prediction. Consider a prediction task over a domain $\mathcal{X}$ and label space $\mathcal{Y}$. The prediction interface $\mathcal{A}$ has access to a private dataset $S = (x_i, y_i)_{i=1}^n \in (\mathcal{X} \times \mathcal{Y})^n$, which outputs a value $a \in \mathcal{Y}$ if given a query $q \in \mathcal{X}$. We denote by Q a query generating algorithm that can adaptively generate a query given the previous released outputs. Namely, we denote by $\mathcal{A}(S) \rightleftharpoons_T Q = (q_t, a_t)_{t=1}^T$ the sequence of query-response pairs generated by the prediction interface $\mathcal{A}$ over a sequence of length T queries on dataset S, where $a_t = \mathcal{A}_t(a_1, ..., a_{t-1}, S, q_t)$.

The privacy guarantee of private prediction is applied for a sequence of predictions generated by the interface $\mathcal{A}$.

Definition 2.3 (Privacy-preserving prediction interface)**.** [Dwork and Feldman, 2018] A prediction interface $\mathcal{A}$ is (ϵ, δ)-differentially private, if for every interactive query generating algorithm Q, the output $\mathcal{A}(S) \rightleftharpoons_T Q = (q_t, a_t)_{t=1}^T$ is (ϵ, δ)-DP with respect to dataset S.

Privacy-preserving prediction algorithms can be useful in a variety of situations where releasing a DP model is restricted or not practical. For example, companies that train a privacy-preserving model and only require making a limited number of predictions can rely on a prediction interface instead of releasing the entire model. In addition, in health or financial data scenarios, private prediction algorithms allow for a cloud-based interface to be exposed, which can also help to ensure compliance with regulatory requirements.

Individual RDP. Our privacy analysis relies on individual privacy loss, which accounts for the maximum possible impact of an individual data point on a dataset. The following definition states the individual privacy loss in terms of Rényi divergence.

Definition 2.4 (Individual RDP [Feldman and Zrnic, 2021])**.** Fix $n \in \mathcal{N}$ and a private data point $z = (x, y) \in \mathcal{X} \times \mathcal{Y}$. We say that a randomized algorithm $\mathcal{A}$ satisfies (α, ρ)-individual Rényi differential privacy for z if for all datasets $S = (z_1, ..., z_m)$ such that $m \leq n$ and $z_i = z$ for some i, it holds that

$$D_\alpha^{\leftrightarrow}\big(\mathcal{A}(S) || \mathcal{A}(S^{-i})\big) \leq \rho,$$

where $D_\alpha^{\leftrightarrow}$ denotes the max of $D_\alpha\big(\mathcal{A}(S) \| \mathcal{A}(S^{-i})\big)$ and $D_\alpha\big(\mathcal{A}(S^{-i}) \| \mathcal{A}(S)\big)$.

Note that the individual RDP parameter ρ is a function of a data point z, and thus does not imply the standard RDP guarantee in Definition 2.2. However, we can obtain the standard RDP guarantee by requiring that all data points z satisfy individual RDP with the same ρ.

Now, we present an example of individual RDP computation on Gaussian mechanism.

Lemma 2.5 (Linear queries with Gaussian mechanism [Feldman and Zrnic, 2021])**.** *Let $S = (z_1, ..., z_n) \in (\mathcal{X} \times \mathcal{Y})^n$. Suppose that $\mathcal{A}$ is a d-dimensional linear query with Gaussian noise addition, $\mathcal{A}(S) = \sum_{j \in [n]} q(z_j) + \mathcal{N}(0, \sigma^2 \mathbf{1}_d)$ for some $q : \mathcal{X} \times \mathcal{Y} \to \mathcal{R}^d$. Then $\mathcal{A}$ satisfies*

$$D_\alpha^{\leftrightarrow}\big(\mathcal{A}(S) || \mathcal{A}(S^{-i})\big) \leq \frac{\alpha ||q(z_i)||_2^2}{2\sigma^2}$$

individual RDP for z_i. Note that by replacing $||q(z_i)||_2$ with the ℓ_2 global sensitivity of $q(\cdot)$, the expression above recovers the standard RDP of Gaussian mechanism.

The following theorem states the composition property of individual privacy. For a sequence of algorithms, as long as the composition of individual RDP parameters does not exceed a pre-specified budget for all data points, the output of the adaptive composition preserves the standard RDP guarantee.

Theorem 2.6 (Corollary 3.3 [Feldman and Zrnic, 2021])**.** *Fix any $G \geq 0$ and any $\alpha \geq 1$. For any input dataset $S =$*

$(z_1, ..., z_n)$ *and for any sequence of algorithms* $\mathcal{A}_1, ..., \mathcal{A}_T$*, let* $\rho_t^{(i)}$ *denote the individual RDP parameter of the* t*-th adaptively composed algorithm* $\mathcal{A}_t$ *with respect to* z_i*. if* $\sum_{t=1}^{T} \rho_t^{(i)} \leq G$ *holds almost surely for all* $i \in [n]$ *then the adaptive composition* $\mathcal{A}^{(T)}$ *satisfies* (α, G)*-RDP.*

The composition rule described above is known as fully adaptive composition [Rogers et al., 2016], which takes *adaptively-chosen* privacy parameters instead of pre-specified ones in the classical adaptive composition. This type of composition is necessary for individual privacy since the individual RDP parameters themselves are random variables that depend on the outputs released by previous composed mechanisms.

To implement the composition above, we need a tool called *Rényi filter*, which is designed to ensure that the composed individual privacy parameters is maintained within a given budget G for all individuals. In practice, we can implement Rényi filter by providing each data point with an individual accountant that estimates its composed individual RDP $\sum_{t=1}^{T} \rho_t^{(i)}$ and dropping the data point once it exceeds the budget, as shown in Algorithm 1.

However, despite its tighter privacy analysis, this technique has been criticized for its computational cost of tracking individual privacy costs for all data samples. In this work, we demonstrate that KNN works seamlessly with the individual RDP accountant. Only selected neighbors are required to update their individual privacy accountants, which significantly reduces the computational cost.

Algorithm 1 Adaptive composition $\mathcal{A}^{(T)}$ with Rényi filter

Input: Dataset $S \in (\mathcal{X} \times \mathcal{Y})^n$, sequence of algorithms $\mathcal{A}_{1:T}$ and privacy budget G.
for $t = 1, ..., T$ **do**
  For all $z_i \in S$, compute $\rho_t^{(i)} = \sup_{S' \in \mathcal{S}} D_\alpha^{\leftrightarrow} \left(\mathcal{A}(a_{1:t-1}, S') || \mathcal{A}(a_{1:t-1}, S'^{-i})\right)$
  Update the active set $S = \{z_i | \sum_{j=1}^{t} \rho_j^{(i)} \leq G\}$
  Compute $a_t = \mathcal{A}_t(a_{1:t-1}, S)$
end for
Return $(a_1, ..., a_T)$

3 PRIVATE PREDICTION WITH IND-KNN

To overcome the limitations of private training, we propose *Individual Kernelized Nearest Neighbor* (Ind-KNN)—a k-nearest neighbor-based private prediction algorithm that achieves a comparable DP guarantee and test accuracy to that of private training.

Notations and setup. We focus on the task of multi-class classification. Given a private dataset $S = (x_i, y_i)_{i=1}^n \in (\mathcal{X} \times \mathcal{Y})$, we assume y_i is an one-hot vector over c class, i.e., $y_i \in \{0, 1\}^c$. Let $\phi(\cdot)$ denote a *public* feature extractor that maps the input $x \in \mathcal{X}$ to a fixed-length feature representation $\phi(x) \in \mathcal{R}^d$. This could be image features extracted from the penultimate layer of a ResNet50 pre-trained model or language features extracted from the final layer of a transformer model. The feature extractor is used to encode both the private dataset and public queries.

Algorithm 2 Privacy-preserving prediction with naive kNN

Input: Dataset $S \in (\mathcal{X} \times \mathcal{Y})^n$, sequence of queries $q_1, ..., q_T$, number of neighbor k and the noisy scale σ.
for $t = 1$ to T **do**
  $\mathcal{N}_k :=$ top k nearest neighbors of the query q_t
  $a_t = \arg\max_{j \in [c]} \left(\sum_{i \in \mathcal{N}_k} y_i + \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{1}_c)\right)_j$
end for
Return $(a_1, ..., a_T)$

Previously, k-Nearest Neighbor (kNN) has been used for privacy-preserving prediction by Zhu et al. [2020] (Algorithm 2). In this method, when a query q_t arrives, the top k nearest neighbors are selected from the private dataset based on the distance in the feature space, and their labels are utilized for prediction through a Gaussian mechanism.

However, the privacy loss of Algorithm 2 accumulates rapidly as the number of queries increases, owing to its conservative privacy analysis that bounds the worst-case individual privacy loss over all individuals. In contrast, the Ind-KNN approach emphasizes individual privacy accounting, providing precise control over privacy loss at an individual data level. This allows each data point's privacy to be charged by the exact amount of its contribution to the query response, and private data is removed once its own privacy budget has been exhausted.

We propose a novel solution *Individualized Kernelized Nearest Neighbor* (Ind-KNN) in Algorithm 3. Intuitively, nearest neighbor-based prediction leak little to no private information when the query point is near a dense region of the training data. This is because the result of the query is determined by a large number of training samples and hence is insensitive to individual training points. We make several modifications to Private kNN to realize this intuition.

First, we introduce individual privacy accounting by assigning each private data point (x_i, y_i) with a pre-determined privacy budget B, represented by the variable $z_i := B$. For each query, the algorithm updates the private dataset S to only include data points where $z_i > \frac{1}{2\sigma_1^2}$. This ensures that the privacy budget for each individual is not exceeded.

Second, Ind-KNN improves upon Algorithm 2 by utilizing a pre-specified threshold τ and a kernel-based similarity function $\kappa(\cdot, \cdot)$ to select only neighbors with similarity above τ. This approach allows only the selected neighbors to be

Algorithm 3 Kernelized-nearest-neighbor with individual privacy accounting (Ind-KNN)

1: **Input**: Dataset $S \in (\mathcal{X} \times \mathcal{Y})^n$, the kernel function $\kappa(\cdot,\cdot)$, the threshold τ, sequence of queries $q_{1:T}$, the noisy scale σ_1, σ_2 and the individual budget B.
2: Initialize individual budget $z_i = B, \forall i \in [n]$.
3: **for** $t = 1$ to T **do**
4: Update the active set $S = \{(x_i, y_i) | z_i \geq \frac{1}{2\sigma_1^2}\}$.
5: Release the number of selected neighbors: $K_t := \sum_{(x_i,y_i)\in S} \mathbb{I}[\kappa(x_i, q_t) \geq \tau] + \mathcal{N}(0, \sigma_1^2)$.
6: **for** $(x_i, y_i) \in S$ **do**
7: Update the remaining budget z_i after releasing K_t: $z_i = z_i - \frac{1}{2\sigma_1^2} \cdot \mathbb{I}[\kappa(x_i, q_t) \geq \tau]$.
8: Evaluate individual contribution $f_t : \mathcal{X} \times \mathcal{Y} \to \mathcal{R}^c$ as $f_t(x_i, y_i) := \min\left(\kappa(x_i, q_t) \cdot y_i \cdot \mathbb{I}[\kappa(x_i, q_t) \geq \tau], \sigma_2\sqrt{2K_t \cdot z_i} \cdot \mathbf{1}_c\right)$
9: Update the remaining budget z_i after releasing label: $z_i = z_i - \frac{||f_t(x_i,y_i)||_2^2}{2\sigma_2^2 \cdot K_t}$.
10: **end for**
11: $a_t = \arg\max_{j\in[c]} \left(\sum_{(x_i,y_i)\in S} f_t(x_i, y_i) + \mathcal{N}(\mathbf{0}, \sigma_2^2 \cdot K_t \cdot \mathbf{1}_c)\right)_j$.
12: **end for**
13: **Return** $(a_1, ..., a_T)$

accountable for their privacy loss, preserving the privacy budget of un-selected private individuals for future queries. It is worth noting that simply selecting the exact top k neighbors, as in Algorithm 2, is not consistent with individual privacy loss. This is because the decision of selection is dependent on the dataset: a $k+1$th nearest neighbor in one dataset may be the top nearest neighbor in another dataset. Hence, all private data points must account for their individual privacy loss, even if only a subset of them contribute to the prediction, according to the definition of individual RDP (Definition 2.4).

Moreover, Ind-KNN employs kernel weights for prediction aggregation instead of equal weight for all nearest neighbors. In our experiments, we consider two types of kernel functions, RBF and cosine, to measure the similarity. For example, the RBF kernel is defined as $\kappa(x, q_t) := e^{\frac{-||\phi(x)-\phi(q_t)||_2^2}{\nu^2}}$, where $\phi(x)$ and $\phi(q_t)$ are the encoded feature and ν is a scalar parameter. This adaptation, made possible by individual privacy accounting, results in a more accurate characterization of each individual's contribution to the query. However, changing from equal weight to kernel weight in Algorithm 2 would not alter its privacy analysis (as the worst-case kernel weight is bound by 1), but would instead decrease the signal-to-noise ratio (each neighbor's contribution would be less than 1).

Finally, Ind-KNN dynamically adjusts the magnitude of noise added to the noisy prediction by publishing the number of neighbors at each query. We find that adding noise with variance proportional to K_t is crucial for good performance. This allows us to adjust the margin of the voting space — the difference between the largest and the second largest coordinate of $\sum_{(x_i,y_i)\in S} f(x_i, y_i)$ adapted to the noise scale. Specifically, when the margin is significant, adding larger noise will not change the output label, but it reduces each individual's individual privacy loss proportional to the reciprocal of K_t, enabling them to participate in more queries in the future.

Algorithm. The modifications made in Ind-KNN are summarized in Algorithm 3. Specifically, since the number of selected neighbors is considered private information, each selected neighbor accounts for its individual privacy loss due to releasing $\mathbb{I}[\kappa(x_i, q_t) \geq \tau]$ by subtracting z_i with $\frac{1}{2\sigma_1^2}$ at line 7 of Algorithm 3. Meanwhile, $f_t(x_i, y_i)$ at line 8 accounts for the individual contribution of releasing its label associated with kernel weight. The first term represents the "weighted" one-hot label for selected neighbors and all-zero vectors for unselected private data. The second term $\sigma_2\sqrt{2K_t z_i}$ ensures that the incurred individual privacy loss of releasing label will not go beyond the remaining budget z_i.

Lemma 3.1 (Individual RDP of releasing a_t)**.** *Given a query q_t, for each (x_i, y_i), define the function $f_t : \mathcal{X} \times \mathcal{Y} \to \mathcal{R}^c$ as line 8 in Algorithm 3. Then the release of $a_t = \arg\max_{j\in[c]} \left(\sum_{(x_i,y_i)\in S} f_t(x_i, y_i) + \mathcal{N}(0, \sigma_2^2 \cdot K_t \cdot \mathbf{I}_c)\right)_j$ satisfies $(\alpha, \frac{\alpha \cdot ||f_t(x_i,y_i)||_2^2}{2\sigma_2^2 \cdot K_t})$ individual RDP for each (x_i, y_i).*

The proof directly follows from Lemma 2.5 and the post-processing property of individual privacy. Note that for unselected private data, their individual privacy loss is always zero since their individual contribution $f(\cdot)$ is zero.

Theorem 3.2. *Algorithm 3 satisfies $(\alpha, B\alpha)$-RDP for all $\alpha \geq 1$.*

Proof sketch. The proof (deferred to the appendix) makes use of the facts that: (1) the decision rule for "being selected" is not influenced by any other private data points, thus, "unselected" neighbors does not incur any individual privacy loss. (2) adding/removing one selected neighbor would only change $\sum_{(x_i,y_i)\in S} \mathbb{I}[\kappa(x_i, q_t) \geq \tau]$ by one, thus the release of K_t satisfies $(\alpha, \frac{\alpha}{2\sigma_1^2})$ individual RDP for selected neighbors. (3) the release of label associated with the kernel weight satisfies $(\alpha, \frac{\alpha||f_t(x_i,y_i)||_2^2}{2\sigma_2^2 K_t})$ individual RDP. □

Remark 3.3. We remark that the privacy guarantee of Ind-KNN is determined by the given individual budget, and remains the same regardless of the number of predictions made. However, as the number of predictions increase, the exclusion of private data may result in a degradation of the algorithm's utility.

3.1 EFFICIENT IND-KNN

In this section, we present two novel techniques that aim to improve the efficiency of Ind-KNN in terms of both utility and computational cost.

Ind-KNN with prediction reuse. The first technique improves the utility of Ind-KNN by exploiting the previously released predictions. We acknowledge that the query-response pairs that have been disclosed can be considered public information. Therefore, we incorporate those predictions into the active set S without any limitation on their privacy budgets. The results of our experiments demonstrate that this extension effectively mitigates the utility loss caused by the exclusion of private data points and improves the test accuracy when handling a large number of queries.

Ind-KNN with hashing. Algorithm 3 requires searching through all private data to answer each query, which can be computationally expensive if the private dataset is large. To address this issue, we present a variant of Ind-KNN that incorporates locality-sensitive hashing (LSH) [Gionis et al., 1999] for efficient nearest neighbor search. The full algorithm of Ind-KNN-Hash is in the appendix. LSH is a well-established technique to speed up the approximate nearest neighbor search. The principle behind the algorithm is to apply LSH to group private data points into "buckets" based on their hash values. When a query is made, the algorithm only needs to search the bucket that the query falls into, rather than searching through the entire dataset.

Concretely, Ind-KNN-Hash creates L hash tables $\mathcal{F} = (f_1, ..., f_L)$ with each of them maps a feature $\mu \in \mathcal{R}^d$ to a b-dimension bucket. For each table f, the algorithm generates b independent random Gaussian vectors from $\mathcal{N}(\mathbf{0}, \mathbf{1}_d)$, denoted by r_j for $1 \leq j \leq b$. Then we encode μ with $f(\mu) = (h_1(\mu), ..., h_b(\mu))$, where $h_j(\mu) = 0$ if $r_j^\top \mu < 0$, otherwise $h_j(\mu) = 1$. The algorithm then indexes all private data points into the hash tables using their encoded features. When a query q_t is received, the algorithm uses LSH to retrieve a set of private data points that are hashed into the same bucket in at least one table, which is denoted by $\mathcal{F}(q_t)$. Finally, Algorithm 3 is called to label each query with a slight modification on the active set, which is now restricted to the retrieved data points with non-negative individual budgets. Typically, increasing the number of hash tables L and reducing the bucket size b results in more accurate neighbors but higher computational costs.

Incorporating LSH into Ind-KNN does not impose any additional privacy cost. This is because the encoding of each private data point is based on random Gaussian vectors and is executed independently of any other private data points.

4 EXPERIMENTS

We consider the following standard image classification and language classification datasets. For each dataset, we take the training set as the private domain and the testing set as the public domain.

Image classification. We evaluate our method on two widely used image classification benchmarks, CIFAR-10 [Krizhevsky et al., 2009] and Fashion MNIST [Xiao et al., 2017]. For CIFAR-10, we employed the recent Vision Transformer (ViT) model [Dosovitskiy et al., 2020], which is pre-trained on the ImageNet-21k consisting of 14 million images and 21843 classes). The extracted from the ViT model are represented as 768-dimensional vectors. For Fashion MNIST, we consider the publicly available ImageNet-pretrained ResNet50 He et al. [2016] from Pytorch as the feature extractor. The model returns a 1000-dim vector for each input image.

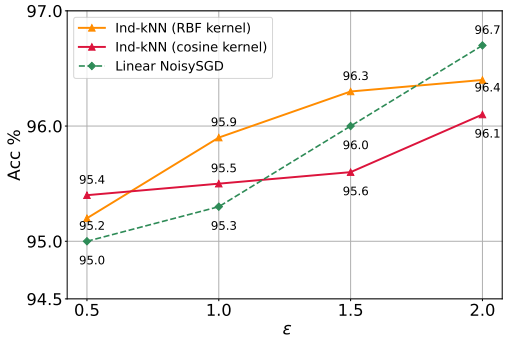
Text classification. We utilize AG News [Zhang et al., 2015] and DBPedia [Lehmann et al., 2015] datasets to evaluate the performance of Ind-KNN on text classification tasks. We employ sentence embedding models [Zhao et al., 2022, Reimers and Gurevych, 2019] to extract features. Specifically, we utilize the `all-roberta-large-v1` sentence-transformer, which has been fine-tuned on a 1B sentence pairs dataset using a self-supervised contrastive learning objective. The extracted features are 1024-dimensional vectors for each text instance.

We consider the following two algorithms for comparisons:

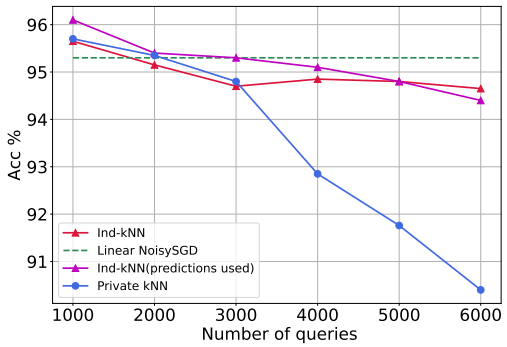
Linear+NoisySGD [Tramèr and Boneh, 2021] is a private training benchmark that has been shown outperforming end-to-end privacy-preserving deep learning methods (including those pre-trained on public data, see De et al. [2022]) for a wide range of ϵ. We consider this algorithm as a reference point for private training to investigate how well Ind-KNN performs compared to private training while we gain those computational savings. We implement the algorithm by training a linear model with features extracted from the same extractor as Ind-KNN. We use the default batch size 256 and clip the gradient norm to 0.1. The model is trained for 10 epochs with a grid search over the learning rate and the noise level is determined by the target privacy budget.

Private-kNN [Zhu et al., 2020] is a private prediction baseline that we consider. For each query, the algorithm first samples a random subset from the private dataset, retrieves the k-nearest neighbors from the subset (based on the extracted features), and then releases the noisy label of kNN prediction using Report-Noisy-Max. We tune the sampling ratio and the number of neighbors on the validation set. The noise scale is calibrated based on the target privacy budget.

Hyper-parameters of Ind-KNN. We set the individual RDP budget B such that using RDP to DP conversion on $(\alpha, B\alpha)$-RDP satisfies the predefined privacy budget (ϵ, δ). Then, we set the noise scale σ_1 to be $\sqrt{\frac{T}{6B}}$ to use roughly half of the individual RDP budget B for each data point being selected at every query and tune the noise scale σ_2 on



(a) Accuracy of 1000 queries on CIFAR-10.



(b) Accuracy vs number of query on CIFAR-10 under $(1, 10^{-5})$-DP.

Figure 1: Privacy-utility trade-offs on CIFAR-10. We plot the median accuracy across 5 independent runs.

the validation set. We consider two kernel methods, the RBF kernel $\kappa(x, q) = e^{\frac{-||\phi(x)-\phi(q)||_2^2}{\nu^2}}$ and the cosine similarity $\kappa(x, q) = \cos(\phi(x), \phi(q))$. A linear scaling search is run on the minimum kernel weight threshold τ for each kernel method. Additional details are given in the appendix.

Experiment setting. For all experiments, we use a random seed to generate a validation set of size T. For example, we randomly sample 1000 examples from the CIFAR-10 testing dataset and tune the best hyper-parameters of all approaches on the validation set. We then report the median accuracy across 5 independent sampled query sets. All experiments are conducted on a server with an Intel i7-5930K CPU @ 3.50GHz and Nvidia TITAN Xp GPU.

4.1 MAIN RESULTS

Privacy-accuracy trade-off on CIFAR-10. In the top figure of Figure 1, we plot the median accuracy evaluated on 1000 randomly chosen queries from the CIFAR-10 test set over a range of privacy budget ϵ. The hyper-parameters were fine-tuned for each algorithm at each value of ϵ. For Ind-kNN, we found that the best hyper-parameter τ (the minimum threshold) increases as the privacy budget grows. We note this because, with smaller value of ϵ, the added noise requires a larger margin among the selected neighbors' votes to determine the correct output. This larger margin, in turn, corresponds to a smaller threshold and more selected neighbors. For Ind-KNN with RBF kernel, we set the kernel bandwidth to $\nu = e^{1.5}$ and search for the optimal minimum threshold τ on the validation set. We find that different choices of kernel bandwidth in the RBF kernel produce similar accuracy results. As shown in Figure 1, Ind-kNN with RBF kernel performs slightly better than its cosine kernel and both kernel methods are comparable to Linear NoisySGD across various value of ϵ.

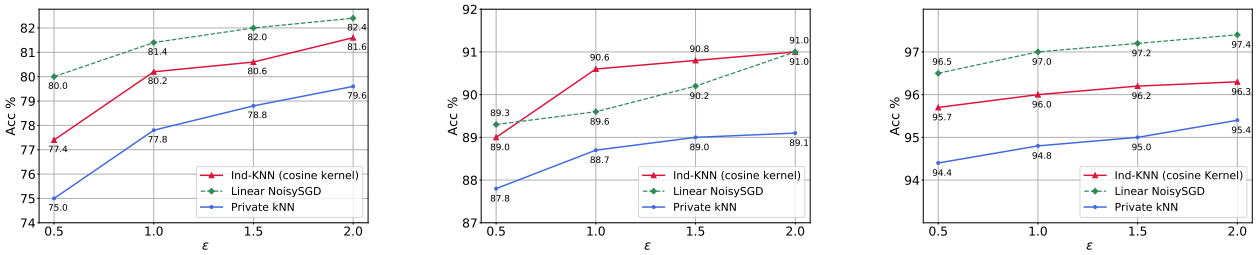
Accuracy vs number of queries on CIFAR-10. Given a fixed privacy budget, the accuracy of all private prediction methods typically degrades as the number of predictions increases, while the accuracy of private training methods remains unaffected. In the bottom figure of Figure 1, we study how quickly the accuracy of Ind-KNN drops as the number of queries increases. We present the median accuracy of answering T queries over five independent rounds. The accuracy of Private kNN drops rapidly with the increasing number of queries. This decline is expected, as Private kNN applies the standard Rényi composition theorem to analyze privacy loss, requiring the noise level to increase proportionally to the square root of T. In contrast, Ind-KNN uses individual privacy accountants, which only require selected neighbors to account for privacy loss, resulting in no significant accuracy drop as more queries are answered. Furthermore, exploiting released predictions allows Ind-KNN to answer an additional 1000 queries (from $T = 2000$ to $T = 3000$) without an accuracy drop. The figure also shows that if the number of queries is less than 2000, Ind-KNN can in fact outperform Linear NoisySGD, making it a practical alternative to private training methods when only a small number of predictions is needed.

Privacy-accuracy trade-off on Fashion MNIST, AG News and DBPedia. Next, we examine the privacy-accuracy trade-off on Fashion MNIST, AG News and DBPedia datasets. We use Ind-KNN with cosine kernel for all datasets. Figure 2a shows that Ind-KNN outperforms Private-kNN for all values of the privacy parameter ϵ on Fashion MNIST. On AG News, we compare the performance of Ind-KNN to that of Linear NoisySGD, and the results are presented in Figure 2b. We evaluate $T = 800$ queries on AG News and find that the accuracy of Ind-KNN either surpasses or matches that of Linear NoisySGD for $\epsilon \geq 0.5$. We also observe similar improvements over Private-kNN on DBPedia.

Overall, Ind-KNN demonstrates its versatility by delivering competitive accuracy results on all three datasets, making it a promising solution for balancing differential privacy and accuracy.

4.2 ABLATION STUDIES

We first perform an ablation study in Figure 3 to better understand how the periodical retraining affects the performance of private training method and our Ind-KNN in terms of computational and privacy cost on CIFAR-10.



(a) Accuracy of 500 queries on FMNIST. (b) Accuracy of 800 queries on AG News. (c) Accuracy of 800 queries on DBPedia.

Figure 2: Privacy-accuracy trade-offs on FMNIST, AG News and DBPedia. We consider $\delta = 10^{-5}$ for FMNIST and AG News and $\delta = 10^{-6}$ for DBPedia.

Periodical retraining. In Figure 3a, we provide empirical measurements of the amortized computational cost associated with periodical retraining on CIFAR-10 of answering a stream of total $T = 10^5$ queries. We assume a retraining request is triggered every time the model has answered Q queries. To simplify the analysis, we assume each retraining is performed on the same dataset. For Linear NoisySGD, we retrain the model for 10 epochs and we calculate the per-query computational cost by dividing the total time spent on retraining and answering T queries by T. This provides an estimate of the average time required to answer a single query. For Ind-KNN with the cosine kernel, the average time of making predictions with is reported. Ind-KNN-Hash uses 30 hash tables with the width parameter $b = 8$. Our results demonstrate that the computational cost per query remains constant for Ind-KNN and Ind-KNN-Hash, as they do not require retraining the model, and the time required to add or delete individual data points is negligible. In contrast, for Linear NoisySGD, every retraining request incurs a substantial computational cost and the privacy loss grows $\propto \frac{1}{\sqrt{Q}}$ (proportional to the square root of total epochs). These findings highlight the advantage of Ind-KNN and Ind-KNN-Hash over Linear NoisySGD in terms of efficiency and resource utilization for machine unlearning and other scenarios with periodic retraining requests.

Figure 3b evaluates the accumulated privacy loss of answering a stream of $T = 2000$ queries on CIFAR-10. We tune hyper-parameters for both approaches such that the averaged accuracy of answering T queries is aligned to 96.0%. We consider two types of retraining scenarios: $Q = 100$ and $Q = 200$. Periodic retraining has a negligible privacy impact on Ind-KNN. Therefore, we only use one red curve to indicate the privacy loss of Ind-KNN under two scenarios. The individual privacy budget of Ind-KNN is pre-determined, thus the standard privacy guarantee remained unchanged when making more predictions. The yellow curve plots the median of individual privacy loss over all private data points and reflects how much individual privacy loss deteriorates as the number of answered queries increases. We note the median individual privacy loss is $\epsilon = 1.2$ after answering 2000 queries, which suggests that only half of the privacy budget has been spent at an individual level. The privacy loss curve of Ind-KNN and two Linear NoisySGDs are met when there received six retraining requests. This suggests that if there are more than six retraining requests among the 2000 queries, the privacy loss of Ind-KNN would be better than that of Linear NoisySGD.

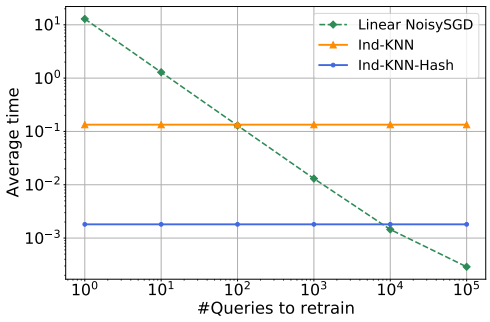
Table 2: Test Accuracy of $T = 1000$ queries on CIFAR-10 under different pre-trained models: vision transformer (ViT) [Dosovitskiy et al., 2020], SimCLRv2 model [Chen et al., 2020] and ResNet50 [He et al., 2016].

$\epsilon(\delta = 10^{-5})$	Method	ResNet50	SimCLRv2	ViT
$\epsilon = 0.5$	Linear NoisySGD	86.2%	89.7%	95.0%
	Private kNN	73.1%	76.0%	94.4%
	Ind-kNN	79.4%	82.4%	95.2%
$\epsilon = 2.0$	Linear NoisySGD	88.4%	90.2%	96.7%
	Private kNN	81.6%	84.7%	96.3%
	Ind-kNN	82.8%	86.3%	96.4%
$\epsilon = \text{inf}$	Linear NoisySGD	90.0%	90.7%	97.0%
	Private kNN	82.9%	85.1%	96.6%
	Ind-kNN	84.7%	89.2%	96.9%

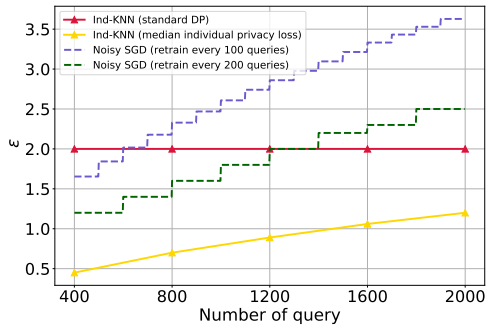
Table 3: The Averaged time (in second) to answer each query on CIFAR-10 and AG News using Ind-KNN and its hashing variants.

Dataset	Table=10	Table=20	Table=30	Ind-KNN
CIFAR-10	0.02	0.03	0.04	0.25
AG News	0.01	0.02	0.03	0.29

Ablation study on hashing. In Sec 3.1, we introduce hashing to improve the computational efficiency of Ind-KNN. We now investigate the trade-off between computational cost and utility of Ind-KNN-Hash on CIFAR-10 and AG News. We set the width parameter $b = 8$ for CIFAR-10 and $b = 9$ for AG News, and evaluate the performance of Ind-KNN-Hash with varying numbers of hash tables. As shown in Figure 4 and Table 3, the accuracy of hashing variants increases with more hash tables and more computational cost. Notably, the computational cost roughly grows linearly with the number of the hash table. In particular, Ind-KNN with 30 hash tables matches the accuracy of the original Ind-KNN for a wide range of epsilon on CIFAR-10 but reduces the running time per query from 0.25 second to 0.03 second. Figure 2b shows similar observations for AG News.

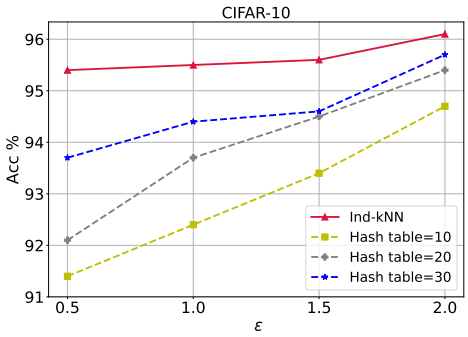


(a) Amortized computational cost vs retraining frequency.

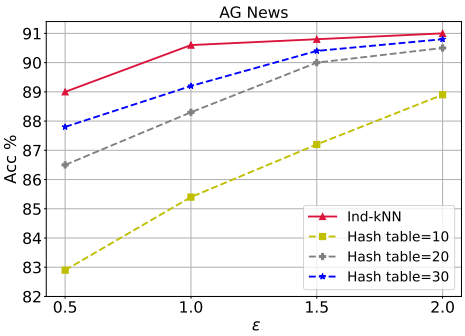


(b) Privacy cost vs |queries| when accuracy is aligned.

Figure 3: (a): We estimate the amortized computational cost by averaging the time (in seconds) spent to answer each query under different retrain settings on CIFAR-10. The x-axis denotes the retraining frequency, i.e., retraining a model every receiving Q queries. (b): The accumulated privacy cost of answering a stream of $T = 2000$ queries when the final accuracy (over 2000 queries) is aligned to 96.0% on CIFAR-10. The red curve fixed the individual privacy budget at the beginning, resulting in a constant privacy loss. The yellow curve reports the median of individual privacy loss across all private data.



(a) Accuracy of $T = 1000$ queries on CIFAR-10.



(b) Accuracy of $T = 1000$ queries on AG News.

Figure 4: Ablation study on hashing under $\delta = 10^{-5}$.

Ablation study on the feature extractor. The quality of the feature extractor plays an crucial role in all three pre-trained feature-based methods. Remarkably, With the ViT feature extractor, even the Private kNN achieves an impressive accuracy of 96.3% at $\epsilon = 2.0$ on CIFAR-10, surpassing the previously reported best result of 95.4% [De et al., 2022] achieved using Wide-ResNets. Next, we present an ablation study focusing on three feature extractors and investigate the efficiency of each method on the CIFAR-10 task. Specifically, we consider three widely used vision models: vision transformer (ViT) [Dosovitskiy et al., 2020], the SimCLRv2 model [Chen et al., 2020] and ResNet 50 [He et al., 2016]. The SimCLRv2-based feature extractor has been considered by prior work Linear NoisySGD (Tramèr and Boneh [2021]), which trains a ResNet model on unlabeled ImageNet using SimCLRv2 model and provides a 4096-dim feature for each input image. For Resnet50, we consider the publicly-available ImageNet-pretrained Reset50 from Pytorch, which achieves a non-private accuracy at 90.0% for LinearSGD. As shown in Table 2, we find that Linear NoisySGD outperforms Private kNN and our Ind-kNN across ResNet50 and SimCLRv2. However, the performance gap decreases when applying a better feature extractor. This can be explained by the fact of their non-private performance. We also note that Private kNN is more fragile when ϵ is small, which could be due to its "loose" privacy analysis. Meanwhile, Ind-kNN handle the setting of small ϵ nicely, and can sometimes outperform Linear NoisySGD with a good feature extractor.

5 SUMMARY

The paper proposes a new algorithm, Individual Kernelized Nearest Neighbor (Ind-KNN), for private prediction in machine learning that is more flexible and updatable over dataset changes than private training. By modifying the KNN prediction and leveraging individualized privacy accountants, Ind-KNN allows a precise control of privacy at an individual level. Through extensive experimentation on four datasets, we demonstrate that Ind-KNN outperforms prior work Private kNN in terms of privacy and utility trade-offs. Furthermore, Ind-KNN exhibits superior computational efficiency and utility when dealing with frequent data updates, surpassing the private training method.

Acknowledgements

YZ, XZ and YXW are partially supported by NSF Award #2048091. YZ was supported by a Google PhD Fellowship and XZ was supported by UCSB Chancellor's Fellowship.

References

Martín Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS-16)*, pages 308–318. ACM, 2016.

Borja Balle, Gilles Barthe, Marco Gaboardi, Justin Hsu, and Tetsuya Sato. Hypothesis testing interpretations and renyi differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pages 2496–2506. PMLR, 2020.

Raef Bassily, Om Thakkar, and Abhradeep Guha Thakurta. Model-agnostic private learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *The Journal of Machine Learning Research*, 12:1069–1109, 2011.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Yuval Dagan and Vitaly Feldman. Pac learning with stable and private predictions. In *Conference on Learning Theory*, pages 1389–1410. PMLR, 2020.

Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Cynthia Dwork and Vitaly Feldman. Privacy-preserving prediction. In *Conference On Learning Theory*, pages 1693–1702. PMLR, 2018.

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

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Vitaly Feldman and Tijana Zrnic. Individual privacy accounting via a renyi filter. *Advances in Neural Information Processing Systems*, 34:28080–28091, 2021.

Marco Gaboardi, James Honaker, Gary King, Kobbi Nissim, Jonathan Ullman, Salil Vadhan, and Jack Murtagh. Psi (ψ): a private data sharing interface. In *Theory and Practice of Differential Privacy*, New York, NY, 2016 2016. URL `https://arxiv.org/abs/1609.04340`.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32, 2019.

Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

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

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE, 2018.

Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3): 793–826, 2011.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical report*, 2009.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, S. Auer, and Christian Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195, 2015.

Alessandro Mantelero. The eu proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3): 229–235, 2013.

Ilya Mironov. Rényi differential privacy. In *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*, pages 263–275. IEEE, 2017.

Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. *arXiv preprint arXiv:1802.08908*, 2018.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

Ryan M Rogers, Aaron Roth, Jonathan Ullman, and Salil Vadhan. Privacy odometers and filters: Pay-as-you-go composition. *Advances in Neural Information Processing Systems*, 29, 2016.

Florian Tramèr and Dan Boneh. Differentially private learning needs better features (or much more data). In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=YTWGvpFOQD-`.

Laurens van der Maaten and Awni Hannun. The trade-offs of private prediction. *arXiv preprint arXiv:2007.05089*, 2020.

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

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, 2015.

Xuandong Zhao, Zhiguo Yu, Ming Wu, and Lei Li. Compressing sentence representation for semantic retrieval via homomorphic projective distillation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 774–781, May 2022.

Yuqing Zhu, Xiang Yu, Manmohan Chandraker, and Yu-Xiang Wang. Private-knn: Practical differential privacy for computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11854–11862, 2020.