Mixed Differential Privacy in Computer Vision

Aditya Golatkar\textsuperscript{1,2} Alessandro Achille\textsuperscript{1} Yu-Xiang Wang\textsuperscript{1,3}
Aaron Roth\textsuperscript{1,4} Michael Kearns\textsuperscript{1,4} Stefano Soatto\textsuperscript{1,2}
\textsuperscript{1}Amazon AWS AI \textsuperscript{2}UCLA \textsuperscript{3}UCSB \textsuperscript{4}UPenn
adityagolatkar@ucla.edu aachille@amazon.com yuxiangw@cs.ucsb.edu
aaroth@cis.upenn.edu mkearns@cis.upenn.edu soattos@amazon.com

Abstract

We introduce AdaMix, an adaptive differentially private algorithm for training deep neural network classifiers using both private and public image data. While pre-training language models on large public datasets has enabled strong differential privacy (DP) guarantees with minor loss of accuracy, a similar practice yields punishing trade-offs in vision tasks. A few-shot or even zero-shot learning baseline that ignores private data can outperform fine-tuning on a large private dataset. AdaMix incorporates few-shot training, or cross-modal zero-shot learning, on public data prior to private fine-tuning, to improve the trade-off. AdaMix reduces the error increase from the non-private upper bound that ignores private data from 167-311\% of the baseline, on average across 6 datasets, to 68-92\% depending on the desired privacy level selected by the user. AdaMix tackles the trade-off arising in visual classification, whereby the most privacy sensitive data, corresponding to isolated points in representation space, are also critical for high classification accuracy. In addition, AdaMix comes with strong theoretical privacy guarantees and convergence analysis.

1. Introduction

When training a deep neural network for visual classification, it is critical to protect privacy by limiting the amount of information that could be gleaned about any individual training sample. Differential Privacy (DP) \cite{Dwork2006} is a theoretical framework to provide strong guarantees on the maximum amount of information that any attacker can extract about an individual training sample. In particular, DP allows users to select the desired trade-off between privacy and accuracy, mediated by a privacy parameter $\epsilon$, which typically depends on the application scenario.

Training large machine learning models while guaranteeing strong privacy for each sample is challenging. In practice, however, one often has a pool of data available for which there are no privacy concerns. This could be a synthetic datasets or datasets designed to be available for public use. Such public data are distinct from the private ones, for which we seek strong privacy guarantees.\footnote{Note that here public data is not the same as data from public sources, as the latter may still require privacy guarantees.} In particular, using large amounts of generic public data for pre-training has recently enabled the creation of language models that achieve DP on the target task while remaining close to state-of-the-art performance \cite{NEURIPS2021_48aa9d75,NEURIPS2021_5b019046}. The same strategy in vision \cite{DBLP:journals/corr/abs-2106-13470}, however, still yields punishing error increases (Tab. 1) ranging from 311\% to 167\% on average across datasets, depending on the desired level of privacy.

Can we, then, make use of public data that is better suited for vision, so we can retain close-to-paragon performance while ensuring privacy?

One sure way of preserving privacy is not to use private data altogether, and recent literature has given us multiple ways to do so. For example, using zero-shot learning \cite{ViTransformer,ZeroShotClassification,WatershedClassification,FineGrainedClassification,ZeroShotEvaluation} one can leverage public data from a different modality (e.g., text) to train a visual model without ever seeing the private data. More generally, one can source or synthesize a few samples of labeled public data from the task distribution, and train using few-shot learning \cite{Qiao2020}, still without using private data. Surprisingly, simple few-shot techniques outperform private training on much larger dataset (Fig. 2, left).

Of course, there may be subtle domain shifts between the public and private data, so ignoring the latter is not a desirable strategy to maintain privacy. The question, then, is how to use even small amounts of public data, in addition to the private data, to break the trade-off between privacy and accuracy. To do so, we change the setting from most work on DP to include labeled public data sourced with the same labels as the target task. We call this setting \textit{mixed differential privacy}, or MixDP.

To address MixDP, we propose to use the public data not just for pre-training the backbone, but for few-shot or zero-shot learning of a classifier on the target tasks, prior to private fine-tuning (Sec. 3.2). In Tab. 1, we show that indeed in
the MixDP setting it is possible to achieve significant gains compared to training with only the private or only the public data, even using a small amount of the latter. To do so, we have to modify existing DP training algorithms to the mixed setting, which leads us to the development of AdaMix, a method for MixDP that uses public data to tune and adapt all major steps of private training, in particular model initialization, clipping of the gradients, and projection onto a lower-dimensional sub space. Some of these ideas to use auxiliary public data were applied in different contexts in isolation, but have suboptimal privacy/performance trade-offs for visual tasks.

In visual classification tasks, the long-tails of the data play an important role in achieving high classification performance [17, 18]. However, DP is a worst-case framework which is highly-influenced by outliers or long-tails. MixDP eases the problem, since it allows to collect public data to ensure that each sub-population is sufficiently covered. This reduced the cost of privacy for the long tails, as we show using a per-instance DP (pDP) analysis of [55].

Unlike most related work, we use NoisyGD rather than DP-SGD as we find that it is faster and more private for our use case. We give a tight analysis of its \((\epsilon, \delta)\)-DP properties using the Analytical Gaussian mechanism [5, 14] (Sec. 3.1). In addition, we present a convergence proof for our algorithm together with new stronger bound for the strongly convex case. This also allows us to describe the utility of the algorithm relative to its non-private counterpart.

To summarize, our contributions are: (1) we introduce AdaMix, a state-of-the-art adaptive method for mixed privacy learning; (2) we show significant improvement w.r.t. baselines on a novel benchmark for MixDP vision tasks; (3) we show that the zero-shot text information can improve the performance of private models (Sec. 4); (4) we analyze the goodness of our method both theoretically (new theorems) and empirically (pDP analysis).

2. Related Work

Most work on Differential Privacy [6, 10, 22, 24, 32, 44, 47, 53, 57] uses public data either for: generic pre-training unrelated to the task [49], or to tune parameters [25, 58, 60], or as additional unlabeled data [37, 38]. Instead, we use a small amount of labeled public data related to the task to improve the accuracy of a private model under a given privacy parameter \(\epsilon\).

Unlabeled public data. PATE [37, 38, 50] trains teacher models on different partitions of the data to predict labels, and privately vote on how to label public (unlabelled) data. Then, a student model is trained in semi-supervised fashion on public data, now partially labelled using the private data. This strategy does not directly allow to train a model on a mix of labeled public and private data. [62] noted that the number of partitions required for meaningful privacy guarantees may be prohibitive for vision datasets. As an alternative, they suggest labeling with a differentially private KNN. In MixDP, where we assume that the public data is already labeled, these algorithms would perform at best similarly to our Only-Public baseline. Instead, what we are interested in effectively using labeled private data to improve performance relative to already labeled public data.

**DP for Language models.** [34, 57] show that a large language model pre-trained on generic public data can be fine-tuned on task-specific private data with only modest loss in accuracy. In contrast, our focus is on using small amounts of public data for fine-tuning. We note that in language models strong transfer of information from the pre-training is facilitated by the fact that the output space (a sequence of tokens) is the same for all tasks. Conversely, vision models need to relearn different output spaces based on the task (i.e., the last layer is always trained from scratch) reducing the transfer of information. We show that we can partly recover this advantage by either using an initialization based on public data of the same task, or a multi-modal vision-language model. We use CLIP [41] for zero-shot learning.

Adaptive clipping. Similarly to us, [52] studies DP training with access to a small amount of public data, and use adaptive gradient clipping [3, 4, 51, 59] based on the public gradients. Contrary to us, they first train a private model separately and then a mixed public-private model using the private one as a regularizer with compelling results on tabular data. We found this approach ineffective for computer vision, where usually models are pre-trained on public data. In Sec. 4 we show that AdaMix works better on vision tasks, and is further improved with multi-modal data.

Adaptive projection. [25, 58, 60] suggest improving DP-SGD by projecting the gradients into a lower-dimensional space estimated from public data. However, these methods do not leverage the performance improvements derived from training on public data, which are needed to make vision models viable. Using public data both to train and to estimate the subspace introduces more challenges, since the gradients of the public data and private data will then behave differently. Moreover, while they perform an SVD on the individual public sample gradients, we do the SVD of the total gradient matrix. Unlike the SVD of the matrix of individual sample gradients, this is cheaper to compute at each step.

**Analysis.** [2] studies the theoretical limits of mixed privacy learning, but does not provide practical algorithms. We provide both a method for MixDP and a proof of convergence. Our analysis leverages the results of [28] which shows that a particular non-uniform averaging scheme of SGD converges at an \(O(1/t)\) rate, rather than the classical
O(\log t/t) rate. In the context of differentially private learning, [6] showed that NoisySGD achieves the optimal rates for (convex) empirical risk minimization (ERM) [11], [48] provided the first analysis of NoisyGD in the general convex case. To the best of our knowledge, we are the first to formally establish that the use of an arbitrarily small public data gives provably smaller sample complexity in learning compared to the standard private learning setting.

3. Method

Notation. Let \( z \in \mathcal{Z} \) be a datum, with \( z = (x, y) \) being the feature/label pair; \( \mathcal{Z}^* = \bigcup_{n=0}^{\infty} \mathbb{Z}^n \) is the universe of datasets with arbitrary size. For learning, we have access to a private dataset \( D_{\text{pri}} := \{ z_1, ..., z_{N_{\text{pri}}} \} \) and a public one \( D_{\text{pub}} := \{ \tilde{z}_1, ..., \tilde{z}_{N_{\text{pub}}} \} \). The loss function \( \ell : \mathcal{Z} \times \mathcal{V} \rightarrow \mathbb{R} \) takes data \( z \in \mathcal{Z} \) and "weights" \( w \in \mathcal{W} \subset \mathbb{R}^d \) to yield the total loss on the private and public dataset \( L_{\text{pri}}(w) := \sum_{i=1}^{N_{\text{pri}}} \ell_i(w) \) and \( L_{\text{pub}}(w) := \sum_{j=1}^{N_{\text{pub}}} \ell_j(w) \), with \( \ell_i \) and \( \ell_j \) short-hands for \( \ell(w, z_i) \) and \( \ell(w, \tilde{z}_j) \) respectively. Notice that \( L_{\text{pri}} \) and \( L_{\text{pub}} \) are sums, not averages, of the loss functions. Their gradients are indicated by \( g_{\text{pri}}(w) = \nabla_w \ell_i(w), g_{\text{pub}}(w) = \nabla_w \ell_j(w) \), \( G_{\text{pri}}(w) = \nabla_w L_{\text{pri}}(w) \), \( G_{\text{pub}}(w) = \nabla_w L_{\text{pub}}(w) \). In addition, we say \( N = N_{\text{pri}} + N_{\text{pub}} \) and \( L(w) = L_{\text{pri}}(w) + L_{\text{pub}}(w) \).

The average empirical risk is written as \( \hat{R}(w) := L(w)/N \). When the data are drawn from some distribution \( D \), the risk, i.e., generalization error, \( R(w) := \mathbb{E}_{z \sim D}[\ell(z, w)] \). Our goal is to design a differentially private algorithm that retrieves \( \hat{w} \) which minimizes \( L(\hat{w}) \). Its performance is measured by the excess empirical risk, i.e., \( \hat{R}(\hat{w}) - \min_{w \in \mathcal{W}} \hat{R}(w) \); and by the excess risk \( \hat{R}(\hat{w}) - \min_{w \in \mathcal{W}} R(w) \). In the experiments, the latter is measured by the error on a test set.

3.1. Differential privacy

Two datasets \( D, D' \in \mathcal{Z}^* \) are said to be neighboring, written as \( D \simeq D' \), if we can obtain \( D' \) from \( D \) by adding or removing a single data point.

Definition 1 (Differential privacy [15, 16]). For \( \epsilon > 0 \) and \( \delta \geq 0 \), a randomized algorithm \( M \) is \((\epsilon, \delta)\)-differentially private if for any neighboring datasets \( D \simeq D' \) and any measurable \( S \subseteq \text{range}(M) \),

\[
\Pr[M(D) \in S] \leq e^\epsilon \cdot \Pr[M(D') \in S] + \delta.
\]

In our context, \( M \) is the randomized learning algorithm, and \( \mathcal{Y} \) is the space of model weights. This condition bounds the maximum amount of information that an attacker, having access to the weights of the model, can extract about one of the training samples. The parameter \( \epsilon \) is referred to as the privacy parameter: lower values of \( \epsilon \) carry better privacy guarantees, but also make it harder to train an accurate model, as it reduces the amount of information about the training samples that can be represented via the parameters of the model. The goal of a good DP training algorithm \( A \) is to train an accurate model while satisfying the \((\epsilon, \delta)\)-privacy constraint selected by the user where, ideally, \( \epsilon \approx 1 \) and \( \delta = o(1/n) \).

NoisyGD. NoisyGD is defined in Algorithm 1. We now show that NoisyGD is differentially private and that, in the convex case, it converges to the optimal solution.

Proposition 2 (NoisyGD is \((\epsilon, \delta)\)-DP, informal). Algorithm 1 with parameter \( T, \sigma^2 \) such that \( \rho := \frac{T^2}{2\sigma^2} \) satisfies the \((\epsilon, \delta(\epsilon))\)-DP, where \( \delta(\epsilon) := \Phi\left(\frac{\rho}{\sqrt{2}} - \frac{\epsilon}{\mu}\right) - e^\epsilon \Phi\left(-\frac{\rho}{\sqrt{2}} - \frac{\epsilon}{\mu}\right) \), \( \mu = \sqrt{2\rho} \) and \( \Phi \) is the CDF of the normal distribution.

For any prescribed privacy parameter \((\epsilon, \delta)\), it suffices to choose this \( \rho \) parameter, thus from here onward we will use \( \rho \) as the privacy parameter of interest (note that, asymptotically, \( \rho \sim \min\{\epsilon, \epsilon^2/\log(1/\delta)\} \)).

Mixed private learning. We define the problem of differentially private learning when we have access to an additional public dataset mixed private learning. We say the algorithm \( M \) (now taking two arguments \( D_{\text{pri}} \) and \( D_{\text{pub}} \)) satisfies \((\epsilon, \delta)\)-DP if \( M(\cdot, D_{\text{pub}}) \) satisfies \((\epsilon, \delta)\)-DP (Def. 1) for every fixed \( D_{\text{pub}} \). [2] shows that a small public dataset from the same distribution allows one to privately PAC-learn any classes with bounded VC-dimension, but with an inefficient algorithm. We focus on designing practical DP algorithm that can effectively use both public and private data under the setting of convex empirical risk minimization [11] and convex fine-tuning of deep models [1].

3.2. AdaMix

Model. Following [49], we use a linear logistic regression model trained on top of the last layer features of an ImageNet pre-trained network. This reduces the expressivity of the model compared to training a full network. However, the significant reduction in number of parameters makes it a

---

**Algorithm 1: (Projected) Noisy Gradient Descent**

**Data:** Constraint set \( \mathcal{W} \), initialization \( w_1 \), noise level \( \sigma \), clipping threshold \( \tau \), number of iteration \( T \), weight decay parameters \( \lambda \) and \( w_{\text{ref}} \), learning rates \( \eta_t \).

**Result:** \( w_1, ..., w_{T+1} \)

for \( t = 1, \ldots, T \) do

\[
G_t = \sum_{i=1}^{N_{\text{pri}}} \text{clip}_s(g_i(w_t));
\]

\[
n_t \sim N(0, \sigma^2 \tau^2 I);
\]

\[
w_{t+1} \leftarrow w_t - \eta_t (G_t + n_t + \lambda (w_t - w_{\text{ref}}));
\]

\[
w_{t+1} \leftarrow \Pi_{\mathcal{W}}(w_{t+1});
\]

end
better choice DP algorithms and recent deep networks allow for competitive performance on most tasks even when training only the last layer. We normalize the features before feeding them to the logistic model, which both improves the accuracy and helps us more easily bound the gradients at each step.

**Initialization and regularization.** The choice of initialization is normally of minor concern for logistic regression: since the problem is convex we will converge to a global optimum regardless of the initialization. L2 regularization (weight decay), when employed, is usually centering at 0 by default. However, centering the regularizer is critical in differential privacy, as we prove in the following result:

**Theorem 3** (Convergence of NoisyGD on strongly convex problems). Assume \( \mathcal{W} \) is a convex set satisfying \( \sup_{w \in \mathcal{W}} \|w\| \leq B \). Let \( w_1, \ldots, w_{T+1} \) be the parameter vectors returned by running Algorithm 1 on the following regularized loss \( J(w) = \mathcal{L}(w) + \frac{\lambda}{2} \|w - w_{\text{ref}}\|^2 \) with \( w_1 = w_{\text{ref}} \in \mathcal{W} \), learning rate \( \eta_t = \frac{2}{T(t+1)} \) and \( \lambda = \sqrt{\frac{\rho}{2 \|w^* - w_{\text{ref}}\| L}} \). Then, the ensemble solution:

\[
\tilde{w} = \frac{2}{T(T+1)} \sum_{t=1}^{T} t w_t
\]

obeys the bound

\[
\mathbb{E}[\tilde{R}(\tilde{w})] - \tilde{R}(w^*) \leq \frac{\rho (N+1) \lambda B^2}{L T N} + \sum_{t=1}^{T} d \|g^* - w_{\text{ref}}\|_{\text{Lip}} \sqrt{\frac{4 \rho}{2 \sqrt{p} N}}
\]

for any \( w^* \in \mathcal{W} \), where \( \rho \) is the privacy parameter of the algorithm.

Note that as \( T \to +\infty \) the first term vanishes and the second term matches the information-theoretic limit for privately solving general convex ERM problems when \( w_{\text{ref}} = 0 \) [6]. However, this can improve if we choose \( w_{\text{ref}} \) according to the public data such that \( w_{\text{ref}} \) is closer to \( w^* \) than 0. This motivated us to propose choosing \( w_{\text{ref}} \) by training on the public data. Then, we proceed to fine-tune the model on the concatenation of private and public data, using Noisy-GD while preserving \((e, \delta)\)-DP.

**Provable benefits of public data access.** Next we provide a theoretical guarantee for the above approach under standard statistical learning assumptions.

**Theorem 4** (Generalization with access to public data, informal). Assume that private and public data are drawn i.i.d. from the same distribution \( \mathcal{D} \) and that \( R(w) = \mathbb{E}_{(x, y) \sim \mathcal{D}}[\ell(w, (x, y))] \) is \( c \)-strongly convex in \( \mathcal{W} \). Let \( w_{\text{ref}} \) be obtained by training for one epoch on the public dataset and let \( \tilde{w} \) be as in Thm. 3, then at the limit when \( T \) is sufficiently large, the excess risk obeys

\[
\mathbb{E}[\tilde{R}(\tilde{w})] - \mathcal{R}(w^*) \leq \frac{\sqrt{4 d L^2}}{c \sqrt{N_{\text{pub}} N \sqrt{2 \rho}}} + \text{Gen}(\tilde{w}, N) + \text{Gen}(w^*, N),
\]

where \( w^* = \arg\min_{w \in \mathcal{W}} R(w) \) and \( \text{Gen}(w, N) := \mathbb{E}[R(w) - \mathcal{R}(w)] \) is the expected generalization gap of (a potentially data-dependent) \( w \).

To the best of our knowledge, Theorem 4 is the first demonstrated claim that access to even a very small public dataset can improve private learning in the convex ERM setting. We note that the generalization bound decomposes in a first term, which describes the generalization cost incurred by using DP, and a second part which is common to non-private learners. Interestingly, in the Appendix we show that in MixDP the cost of privacy asymptotically vanishes even if the percentage of public data becomes negligible (i.e., when \( N_{\text{pub}} \to \infty \), \( N_{\text{pub}} / N \to 0 \)).

Besides providing a good initialization, access to a small public dataset also enables ways to further improve Noisy-GD in practice, as we will now see.

**Adaptive threshold.** Since the norm of the gradients changes significantly during training, a fixed clipping threshold \( \tau \) is suboptimal throughout training. [52] suggests to tune this threshold adaptively using public data. However, the norm of the average gradient of the public sample, may not yield good clipping thresholds for the per-sample gradients. Instead, we find the following adaptive threshold to be more robust:

\[
\tau = \text{quantile}_{90}(\{\|g_{i}^{\text{pub}}(w)\|\}_{i=1}^{N_{\text{pub}}}),
\]

that is, we take the 90-th quantile of the norm of the public gradients, where \( g_{i}^{\text{pub}}(w) = \nabla_w \ell_i(w) \) is the gradient of the \( i \)-th public sample. Then the clipped gradient we use is

\[
\tilde{g}_{i}^{\text{pri}}(w) = \frac{g_{i}^{\text{pri}}(w)}{\|g_{i}^{\text{pri}}(w)\|} \min(\|g_{i}^{\text{pri}}(w)\|, \tau).
\]

This quantile rule is also used by [3], but we differ in the use of public data to determine \( \tau \), rather than allocating part of the privacy-budget for releasing the quantile. It was established in [12] that gradient clipping has the effect of Huberization of the loss function in the GLM cases. The convergence to the optimal solution of a modified objective function is guaranteed if we fix the clipping parameter. Our approach on the adaptive clipping has the effect of selecting the hyperparameter adaptively in a homotopy style algorithm, which shows further empirical benefits.

**Adaptive subspace projection.** From Thm. 4, the utility of a DP algorithm decreases with the number of parameters \( d \) of the model. In view of this, we do not train the whole network but rather use it as a feature extractor and only train a linear model on top. However, for modern vision architectures, even the number of parameters of the final layer is in the order of tens of thousands. Therefore, we use an adaptive mechanism to project private gradients onto a low-dimensional subspace before adding noise. More precisely,
be the total gradient of the public dataset $D_{\text{pub}}$. Since we are using a linear logistic model, $G_{\text{pub}}(w)$ is a $D \times C$ matrix where $D$ is the number of features and $C$ is the number of classes. Consider the SVD decomposition:

$$G_{\text{pub}}(w) = U S V.$$  

We now project each (clipped) private gradient matrix $\hat{g}_{\text{pri}}^i$ on the top $P$ components, and obtain a matrix $\hat{g}_{\text{pri}}^i = U^T g_{\text{pri}}^i$ of size $P \times C$. Other methods projecting the gradient using public data [25, 58, 60] do not compute the SVD of the total gradient matrix, as noted in Sec. 2. Rather, use the vectorized per-sample gradients of the public examples to compute the principal components to project. Since, in vision application, the size of matrix of all per-sample gradients has a much larger size $N \times FC$ (rather than our $F \times C$), several approximations would be required to carry out this strategy.

**Final algorithm.** The final gradient update step at time $t$ is:

$$w_{t+1} \leftarrow w_t - \eta \left( G_{\text{pub}} + U G_{\text{pri}} + \lambda \cdot w_t \right)$$

where $n_t \sim N(0, \sigma^2 \cdot 2 I_C)$. We then train for $T$ steps. The pseudo-code of algorithm is given in Algorithm 2.

**Proposition 5.** Algorithm 2 with parameter $T$, $\sigma^2$ such that $\rho := \frac{\sigma^2}{2 \pi^2}$ satisfies the $(\epsilon, \delta)$-DP of a Gaussian mechanism in Theorem 8 [5] with parameter $\mu = \sqrt{2 \rho}$.

Note that the privacy guarantee is the same as long as the ratio $\sigma / \sqrt{T}$ remains constant, which gives a degree of freedom in choosing $\sigma$ (hence $T$). Higher values of $\sigma$ requires more training steps $T$, but, by Thm. 3, it also ensures better convergence (which we validate this empirically in Sec. 4). Hence, the user should generally pick the largest $\sigma$ allowed by their computational budget.

### 3.3. Textual side information

CLIP [41] is an embedding trained so that the representation of an image is close to the representation of its textual description. This enables zero-shot learning, i.e., initializing an high-performance image classification model using only the name of the labels or a textual description of the classes, which is generally public. AdaMix can be easily adapted to work in this multi-modal MixDP setting, where the public data comes from a different modality. First, we initialize a linear logistic model using the names of the classes: Let $c_i$ be the text associated to the $i$-th class, the weight matrix is [41]:

$$W = \beta [\text{enc}(c_1), \ldots, \text{enc}(c_C)],$$

that is, each row is the normalized encoding of a label scaled by a parameter $\beta$ (we use $\beta = 100$). Starting from this public initialization, we train with AdaMix on image data.

### 4. Results

**Setting.** Unless otherwise specified, we train a linear logistic model on top of the last layer features of a ResNet-50 [21] pre-trained on ImageNet [13]. For NoisyGD, we initialize the logistic weights to zero, fix $\sigma = 20$ and $\delta = 10^{-5}$. We train with $\lambda = 1e - 2$, weight decay centered at zero, learning rate $\eta \in \{0.0005, 0.001, 0.0025, 0.005\}$, and select the best learning rate for each value of $\epsilon$. For AdaMix, we initialize the weights by training on the public data until convergence. We use the Gaussian mechanism implementation of autodp$^2$ to calibrate the parameters of the training algorithm, in particular $T_{\text{max}}$, in order to achieve the desired $(\epsilon, \delta)$-DP requirement.

**Datasets.** We test our algorithm on the following vision classification datasets commonly used in transfer learning (see Tab. 1). We split each training dataset into a “public” dataset consisting of $N$-samples per class (see Tab. 1), and

---

2[https://github.com/yuxiangw/autodp](https://github.com/yuxiangw/autodp)
consider the remaining training data as private. In all cases, the public data is less than 10% of the private data.

**Baselines considered.** We compare AdaMix with the following baselines: (Fully-Private) Ignore the MixDP setting and simply train on all data as if it was private, starting form a zero initialization and using NoisyGD; (Only-Public) Train a standard non-private model using only the few public samples and discarding the private ones; (PPGD) A version of PPSGD [52], but trained with NoisyGD rather than DP-SGD for easier comparison. Unlike AdaMix, PPGD has several hyper-parameters: we manually searched for the best combination to ensure the best comparison; (NGD) An ablated version of AdaMix which uses fixed hyper-parameters rather than the adaptive components.

### 4.1. Experiments

**Mixed Privacy vs Full Privacy.** In Table 1 we see that, perhaps surprisingly, Only-Public is always significantly better than Fully-Private even if it actually throws away most of the available data. This suggests that, in vision, collecting a small amount of public data may be preferred to collecting a much larger amount of private data (Fig. 2). However, we see that AdaMix is able to effectively use both the private and public data at the same time, and achieves a significant improvement even at relatively small value of $\epsilon$ (e.g., 10% improvement on MIT-67 w.r.t. Only-Public). Instead, the closest method to us, PPGD does not significantly improve over the Only-Public baseline.

On the other hand, thanks to the MixDP training we can obtain high-accuracy models even at restrictive privacy settings. We note that AdaMix performs uniformly better than other methods at all privacy regimes. From the ablation of AdaMix, we see that initializing the model with the public data has the largest impact, followed by adaptive clipping and adaptive projection.

**Membership attacks.** Differential privacy is a very strong requirement, and is it often assumed that using higher value of $\epsilon$ may still provide valuable defense to attacks [7–9, 31, 35, 42, 43, 45, 61] even if the theoretical guarantees are weak. Conversely, [23] suggests using attacks to give a lower-bound on the privacy parameter of the algorithm. In Fig. 1 (right) we plot the Area Under Curve of a membership attack executed against models for different values of $\epsilon$. We use a thresholding based membership attack as proposed in [43]. The loss for samples belonging to the private training set is labelled as 0 and test samples are labelled as 1. Then we simply compute the AUC for this. We find that the sorting of the algorithms does not change: algorithms with better theoretical bounds are also more robust to attacks.

**Performance Boost and size of public data.** We hypothesize that, when enough public data is present, the contribution of using private data may be less noticeable. To test this, we define the Performance Boost of a Mixed Privacy method as $\text{PB} = (\text{err}_{\text{public}} - \text{err}_{\text{paragon}})/(\text{err}_{\text{DP}} - \text{err}_{\text{paragon}})$, where $\text{err}_{\text{paragon}}$ is the error obtained by training on non-private model on both public and private data, $\text{err}_{\text{public}}$ is the error obtained by training only on the public data, and $\text{err}_{\text{DP}}$ is the error obtained by using training a mixed privacy model. This measures how well we can learn from the private data in the MixDP setting compared to the non-private upper-bound. In Fig. 2 we plot the performance boost for different sizes of the public set. We observe that the performance boost derived of using mixed privacy is very high when the public data is scarce, disappears when the amount of public data is comparable or larger than the private data.

### Table 1. Mixed privacy, full privacy and AdaMix

| Public Shots | Non-Private (paragon) | Only-Public (baseline) | Fully-Private PPGD [52] AdaMix $\epsilon = 1$ | Fully-Private PPGD [52] AdaMix $\epsilon = 3$ |
|--------------|-----------------------|------------------------|---------------------------------------------|---------------------------------------------|
| Ox. Flowers [36] | 2                     | 12.34                  | 95.12                                      | 81.42                                       |
| CUB-200 [36]   | 2                     | 31.97                  | 96.79                                      | 81.61                                       |
| Stanford Dogs [26] | 2                     | 10.08                  | 33.87                                      | 15.93                                       |
| MIT-67 [40]    | 5                     | 25.17                  | 69.55                                      | 42.99                                       |
| Oxford Pets [39] | 5                     | 7.17                   | 26.02                                      | 12.37                                       |
| Caltech-256 [20] | 5                     | 14.64                  | 60.72                                      | 27.15                                       |

Regarding the influence of the public data, we can observe that the performance boost is very high when the public data is scarce, and decreases when the amount of public data is comparable or larger than the private data.
Multi-modal initialization. In Fig. 4 we compare CLIP models initialized with (1) a random initialization, (2) a zero-shot initialization based on the public label names (Sec. 3.3), and (3) using few public image samples. While the latter performs the best at a given privacy level $\epsilon$, the zero-shot initialization still improves significantly over the random initialization, thus showing that even just small amount of textual information (which is often available) can be used to significantly boost accuracy. Multi-modal models may also learn a better lower-dimensional representation of the data that is more amenable to DP. In Fig. 4 we compare AdaMix using an ImageNet pretrained ResNet-50 and using CLIP. This is not a direct comparison, but we see a significant performance boost by using CLIP at a given privacy level $\epsilon$, suggesting that the move to multi-modal model may be a logical next step for DP.

Effect of differential privacy on individual samples. DP is a worst-case guarantee which may penalize the accuracy of a model on unusual data, or data on the long-tails of the distribution, which however may play an important role for generalization in vision [17, 18]. MixDP may ease the problem, since it allows to collect public data to ensure that each sub-population is sufficiently covered, lessening the accuracy cost of making that data private. We show this effect using the per-instance DP (pDP) analysis of [55], which measures the privacy loss $\epsilon'$ incurred by each individual $z$ during the private training. This allows us to provide a finer analysis of different methods by comparing the histogram of pDP loss $\epsilon$ for each individual in the dataset $D$ on top of the worst case DP bounds, which we do in Fig. 3 (see Appendix for details and theoretical analysis). We observe that, when using AdaMix, the pDP losses of most samples are the same, and is close to the maximum privacy budget $\epsilon = 3$, indicating that we are using the prescribed privacy budget more effectively and more uniformly across samples, thereby improving utility.

Domain shift. Often, the private and public data may come from slightly different domains. To test the robustness of AdaMix to domain shifts, we use the DomainNet dataset, which aims to classify objects in different domains. In particular, we train a model to classify objects in painting, using however public data coming from real pictures (while the private data is in the painting domain). In Fig. 3 (right),

Figure 2. (Left) Trade-off between public and private examples. We show the accuracy reached on MIT-67 using $N$ samples per class (x-axis) when they are public (green) or private (in red for different values of $\epsilon$). Fully private training requires as much as 10 times more samples. This leaves users with a trade-off between collecting more private data or, if possible, a smaller amount of public data. (Right) Performance boost using a private set for different methods. We plot the Performance Boost (PB) for different DP methods as the ratio of public to private samples changes (see text for definition). We see that AdaMix achieves the best performance boost. As expected, when the public set is relatively large, the boost from adding private data decreases.
we show that, even if the public comes from a different domain, AdaMix is still able to use the public data to significantly improve performance of private training.

**Effect of $\sigma$.** Outside of the learning rate $\eta$, the main other sensitive hyper-parameter of AdaMix is the noise variance $\sigma$. From Prop. 3 we expect that the best accuracy will be obtained using a large $\sigma$ and training for a respectively larger amount of epochs $T$. In the Appendix, we plot the test error as a function of $\sigma$ and we see that indeed, a larger $\sigma$ give better results at the expense of longer training time.

## 5. Discussion

We study Mixed Differential Privacy learning in computer vision, and show that in this setting differential privacy can be used to provide high accuracy models while respecting a given privacy parameter. To do this, we introduce AdaMix, an adaptive differential privacy mechanism that uses the public data to initialize and compute, at each step, an optimal subspace and clipping threshold. We also show that multi-modal vision and text can significantly improve accuracy under a privacy constrain.

**Limitations.** In our analysis we train a linear model on pre-trained features, rather than fine-tuning the whole network. This is common in DP, since fine-tuning multiple layers may not significantly improve the accuracy while the additional parameters negatively affects the privacy bounds. This limitation is partially offset by new models that are trained to have generic and transferable last-layer features without the need of additional fine-tuning. We discuss several theoretical bounds that guide us in understanding the behavior of the algorithm. However, these bounds refer to a slightly simplified version of AdaMix (see Appendix for details) and use hyper-parameters that are asymptotically optimal, but may not be practical.

**Conclusions.** Differentially Private models in computer vision often cannot reach the same level of performance as non-private model for realistic privacy parameters. Using AdaMix in the MixDP learning setting, we have shown that, assuming the presence of a small amount of public data, accurate networks can be produced that reach better accuracies than fully private training without compromising the privacy of the data. We hope that further research in this field may help the adoption of private models in more areas of computer vision.
References

[1] Alessandro Achille, Aditya Golatkar, Avinash Ravichandran, Marzia Polito, and Stefano Soatto. Lqf: Linear quadratic fine-tuning. In IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15729–15739, 2021. 3

[2] Noga Alon, Raf Bassily, and Shay Moran. Limits of private learning with access to public data. In Advances in Neural Information Processing Systems (NeurIPS'19), pages 10342–10352, 2019. 2, 3

[3] Galen Andrew, Om Thakkar, H Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. In Advances in Neural Information Processing Systems (NeurIPS'21), 2021. 2, 4

[4] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. Advances in Neural Information Processing Systems, 32:15479–15488, 2019. 2

[5] Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In International Conference on Machine Learning, pages 394–403. PMLR, 2018. 2, 5, 15

[6] Raf Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In Annual Symposium on Foundations of Computer Science, pages 464–473. IEEE, 2014. 2, 3, 4

[7] Daniel Bernau, Philip-William Grassal, Jonas Robl, and Florian Kerschbaum. Assessing differentially private deep learning with membership inference. arXiv preprint arXiv:1912.11328, 2019. 6

[8] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In 28th USENIX Security Symposium (USENIX Security 19), pages 267–284, 2019. 6

[9] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In USENIX Security Symposium, 2021. 6

[10] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In NIPS, volume 8, pages 289–296. Citeseer, 2008. 2

[11] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. Journal of Machine Learning Research, 12(3), 2011. 3

[12] Xiangyi Chen, Steven Z Wu, and Mingyi Hong. Understanding gradient clipping in private sgd: a geometric perspective. Advances in Neural Information Processing Systems, 33, 2020. 4

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009. 5

[14] Jinshuo Dong, Aaron Roth, and Wéijie Su. Gaussian differential privacy. Journal of the Royal Statistical Society, 2021. 2, 15

[15] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Theory of cryptography conference, pages 265–284. Springer, 2006. 1, 3

[16] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci., 9(3-4):211–407, 2014. 3

[17] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, pages 954–959, 2020. 2, 7

[18] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 2881–2891. Curran Associates, Inc., 2020. 2, 7

[19] Andrea Frome, Greg Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc’Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. 2013. 1

[20] G. Griffin, AD. Holub, and Pietro Perona. The caltech 256. 6

[21] 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, pages 770–778, 2016. 5

[22] Roger Iyengar, Joseph P Near, Dawn Song, Om Thakkar, Abhradeep Thakurta, and Lun Wang. Towards practical differentially private convex optimization. In 2019 IEEE Symposium on Security and Privacy (SP), pages 299–316. IEEE, 2019. 2

[23] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 22205–22216. Curran Associates, Inc., 2020. 6

[24] Bargay Jayaraman and Lingxiao Wang. Distributed learning without distress: Privacy-preserving empirical risk minimization. Advances in Neural Information Processing Systems, 2018. 2

[25] Peter Kairouz, Mónica Ribero, Keith Rush, and Abhradeep Thakurta. Fast dimension independent private adagrad on publicly estimated subspaces. arXiv preprint arXiv:2008.06570, 2020. 2, 5

[26] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition, Colorado Springs, CO, June 2011. 6

[27] Tomer Koren and Kfir Y Levy. Fast rates for exp-concave empirical risk minimization. In Neural Information Processing Systems, pages 1477–1485, 2015. 19

[28] Simon Lacoste-Julien, Mark Schmidt, and Francis Bach. A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method. arXiv preprint arXiv:1212.2002, 2012. 2, 16, 17
[56] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 6

[57] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. arXiv preprint arXiv:2110.06500, 2021. 1, 2

[58] Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. In International Conference on Learning Representations, 2021. 2, 5

[59] Xinwei Zhang, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Jinfeng Yi. Understanding clipping for federated learning: Convergence and client-level differential privacy. arXiv preprint arXiv:2106.13673, 2021. 2

[60] Yingxue Zhou, Steven Wu, and Arindam Banerjee. Bypassing the ambient dimension: Private {sgd} with gradient subspace identification. In International Conference on Learning Representations, 2021. 2, 5

[61] Ligeng Zhu and Song Han. Deep leakage from gradients. In Federated learning, pages 17–31. Springer, 2020. 6

[62] 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. 2
A. Supplementary Material

In the supplementary material we perform, (1) Ablation studies analyzing different components of our algorithm (Appendix A.1), (2) tuning the adaptive clipping threshold can further improve AdaMix (A.2), (3) show that using a larger $\sigma$ (training for longer) is better (A.3), (4) provide additional experimental results (A.4), (5) provide details experimental details (A.5, A.6), and (6) proofs for all the theoretical results presented in the paper (Appendix B, C, D).

A.1. Ablation Studies

In Fig. 5, we ablate the two components of our algorithm, namely, Subspace Projection and Adaptive Clipping. We consider the Noisy-GD (NGD) as the baseline and analyze the improvement provided by the two components separately and finally in combination (AdaMix). We observe that adaptive clipping is the key component of the AdaMix algorithm. We show that using the two components together performs the best across multiple datasets. In the next section we show that further tuning the adaptive clipping threshold can improve the performance even more on some datasets. In Fig. 6, we plot the relative reconstruction error for the private gradients on the public gradient subspace and a random subspace. We observe that the reconstruction error on the public gradient subspace is at least half the random subspace throughout training, which suggests the importance of the public gradients for AdaMix.

In all of our experiments in Fig. 7. However for CUB-200 and Caltech-256 tuning it to 75 percentile works better indicating that AdaMix has more scope for improvement. Note, however that using 90 percentile for CUB-200 and Caltech-256 still outperforms all the baselines.

A.2. Effect of AdaMix clip threshold

We plot the effect of adaptive clipping threshold (percentile) on the test error for AdaMix (we use 90 percentile in all of our experiments) in Fig. 7. However for CUB-200 and Caltech-256 tuning it to 75 percentile works better indicating that AdaMix has more scope for improvement. Note, however that using 90 percentile for CUB-200 and Caltech-256 still outperforms all the baselines.

A.3. Longer training with more noise

Higher values of $\sigma$ requires more training steps, however, Theorem 3 shows that it leads to better convergence, which leads to better generalization. In Fig. 8 we see that indeed, at the same level of privacy, using a large of $\sigma$ significantly reduces the test error compared to using a smaller $\sigma$.

A.4. Additional Experiments

In the main paper we presented detailed experimental results on MIT-67, here we present detailed results on all the remaining datasets.
Figure 8. **Larger values of $\sigma$ performs better.** We plot the test accuracy of NGD and AdaMix using different values of $\sigma$ for $\epsilon = 3$. A larger $\sigma$ performs better but needs more training steps thus verifying the claim empirically across multiple datasets.

### A.4.1 Test Error vs Privacy

We show the test error obtained by different methods for different levels of privacy $\epsilon$ and the robustness to membership attack, similar to Fig. 1 for different datasets in Figs. 9 to 13.

Figure 9. Test error vs Privacy and Robustness to Membership Attacks on Oxford-Flowers

Figure 10. Test error vs Privacy and Robustness to Membership Attacks on CUB-200

Figure 11. Test error vs Privacy and Robustness to Membership Attacks on Oxford-Pets

### A.4.2 Per-Instance Privacy

We show the pDP loss for NGD and AdaMix for different datasets in Figs. 14 to 18, similar to Fig. 3 (we use $\epsilon = 3$).

Figure 12. Test error vs Privacy and Robustness to Membership Attacks on Stanford Dogs

Figure 13. Test error vs Privacy and Robustness to Membership Attacks on Caltech-256

Figure 14. Effect of public data on per-instance DP for Oxford Flowers

Figure 15. Effect of public data on per-instance DP for CUB-200

Figure 16. Effect of public data on per-instance DP for Oxford Pets
A.5. Multi-modal Initialization

We show the effect of using different initializations and CLIP features for different datasets in Figs. 19 to 22, similar to Fig. 4 (we use $\epsilon = 3$). We show that for Stanford Dogs and Oxford Pets datasets (Fig. 19 and Fig. 20) which have images which are very similar to images in ImageNet, ResNet-50 trained on ImageNet performs better than CLIP features. However, the trend for the effect of initialization remains the same across all the datasets.

A.6. Experimental Details

We use Auto-DP library for all of our experiments. For ResNet-50 features we use the torchvision version of the ResNet-50 model and for CLIP features we use the model provided.\(^3\)

To create the public and private datasets, we take 2 samples (for Oxford-Flowers and CUB-200) and 5 samples (for MIT-67, Stanford Dogs, Oxford Pets, Caltech-256) from each class as the public dataset and the remaining samples as the private dataset. In this way, the public set is less than 10% of the private set. We repeat all the experiments with 3 random seeds and report its mean and std.

For all the experiments we try multiple values for the learning rate: $\{1e-3, 2.5e-3, 3e-3\}$ for ResNet-50 experiments and $\{1e-6, 5e-7, 1e-7\}$ for CLIP experiments, and report the best values across 3 random seeds. We use $1e-2$ as the $L_2$ regularization coefficient across all the experiments. For subspace projection we project the gradients on a 2000-dimensional subspace for experiments with ResNet-50 features and 500-dimensional subspace for experiments with CLIP features. For the clipping threshold percentile we use a constant value of 90 percentile across all the experiments in the paper. In Fig. 7, we show that further tuning the clipping threshold percentile can improve performance on certain datasets, however, even with a pre-decided value of 90 percentile it still out-performs all the other methods.

We train the logistic model on the public data (few shot data) for 200 epochs (sames as iterations since we use gradient descent) for multiple values of learning rate: $\{1e-1, 5e-2, 1e-2\}$ for ResNet-50 features and $\{1e-4, 5e-5, 1e-5\}$ for CLIP features and choose the best performing model. For the private training, we use $\sigma = 20$ for all the experiments and calculate the number of iterations required for private training using methods provided in Auto-DP. In the experiments we observe that choosing a higher value of $\sigma$ (thus training for more iterations) generally results in better performance.

\(^3\)https://github.com/openai/CLIP
B. Differential privacy basics

In this section, we describe tools we need from differential privacy and use them to prove that Algorithm 2 and Algorithm 3 satisfy a family of differential privacy parameters.

Lemma 6 (Analytical Gaussian mechanism [5]). For a numeric query \( f : \mathcal{X}^* \to \mathbb{R}^d \) over a dataset \( \mathcal{D} \), the randomized algorithm that outputs \( f(\mathcal{D}) + Z \) where \( Z \sim \mathcal{N}(0, \sigma^2 I_d) \) satisfies \((\epsilon, \delta(\epsilon))\)-DP for all \( \epsilon \geq 0 \) and \( \delta(\epsilon) := \Phi(\frac{\epsilon^2}{2} - \frac{\epsilon}{\sigma}) - e^{\frac{\epsilon^2}{2}} \Phi(-\frac{\epsilon}{2} - \frac{\epsilon}{\sigma}) \) with parameter \( \mu := \Delta/\sigma \), where \( \Delta := \Delta_f = \max_{D \sim \mathcal{D}} \| f(D) - f(D') \|_2 \) is the global L2 sensitivity of \( f \) and \( \Phi \) is the CDF function of \( \mathcal{N}(0, 1) \).

The above lemma tightly characterizes the \((\epsilon, \delta)\)-DP guarantee of a single invocation of the Gaussian mechanism, and the following lemma shows that we can use the same result for an adaptive composition of a sequence of Gaussian mechanisms.

Definition 7 (Gaussian Differential privacy [14]). We say a mechanism \( M \) satisfies \( \mu \)-Gaussian differential privacy (GDP), if it satisfies \((\epsilon, \delta(\epsilon))\)-DP for all \( \epsilon \geq 0 \) and \( \delta(\epsilon) \) being that of a single Gaussian mechanism (in Lemma 6) with parameter \( \mu \).

Lemma 8 (Composition of Gaussian mechanisms [14]). The adaptive composition of a sequence of Gaussian mechanisms with noise level \( \sigma_1, \sigma_2, \ldots \) and global L2 sensitivity \( \Delta_1, \Delta_2, \ldots \) satisfies \( \mu \)-GDP with parameter \( \mu = (\sum_i (\Delta_i/\sigma_i)^2)^{1/2} \).

Specifically, the noisy gradient descent (NoisyGD) algorithm (Algorithm 1) we use is a composition of \( T \) Gaussian mechanisms for releasing the gradients and its privacy guarantee is equivalent to that of a single Gaussian mechanism.

Proposition 9. Let \( \nabla f(w) \) be a function of the private dataset with global L2 sensitivity smaller than \( L \) for any \( w \in \mathcal{W} \), then Algorithm 1 with parameter \( T, \sigma^2 \) such that \( \rho := \frac{T^2 L^2}{2\sigma^2} \) satisfies \( \sqrt{2\rho} \)-Gaussian differential privacy.

Proof. The proof follows from Lemma 8 as Algorithm 1 is an adaptive composition of \( T \) Gaussian mechanisms with global sensitivity \( L \).

C. Per-instance differential privacy

In this section, we provide details on the per-instance differential privacy [55] that we used to generate Figure 3. To cleanly define per-instance differential privacy, we first define indistinguishability.

Definition 10 ((\(\epsilon, \delta\))-indistinguishability). We say two distributions \( P, Q \) are \((\epsilon, \delta)\)-indistinguishable if for any measurable set \( S \)

\[
Pr_P[S] \leq e^\epsilon \cdot Pr_Q[S] + \delta \quad \text{and} \quad Pr_Q[S] \leq e^\epsilon \cdot Pr_P[S] + \delta.
\]

Definition 11 ([55]). We say a randomized algorithm \( M \) is \((\epsilon(\cdot), \delta)\)-per-instance differentially private (pDP) for scalar \( \delta \geq 0 \) and function \( \epsilon : Z^* \times Z \to \mathbb{R}_+ \), such that for any dataset \( D \in Z^* \), individual \( z \in Z \), \( M(D) \) and \( M(D \cup \{z\}) \) are \((\epsilon(D, z), \delta)\)-indistinguishable.

pDP loss \( \epsilon \) is a function of one particular pair of neighboring datasets. It describes the formal privacy loss incurred to the particular individual \( z \) that is added (if \( z \) is not part of the input dataset) or removed (if \( z \) is part of the input dataset). pDP is a strict generalization of DP, as we can recover DP from pDP by maximizing \( \epsilon(\cdot) \) over \( D, z \).

Lemma 12. If \( M \) is \((\epsilon(\cdot), \delta)\)-pDP, then \( M \) is also \((\sup_{D \in Z^*, z \in Z} \epsilon(D, z), \delta)\)-DP.

We emphasize that the pDP loss \( \epsilon(\cdot) \) itself is data-independent, but specific evaluations of the pDP losses (e.g., \( \epsilon(D_{-z}, z) \) or \( \epsilon(D, z) \)) depends on the private dataset \( D \), thus should not be revealed unless additional care is taken to privately release these numbers.

For our purpose, we are interested in the distribution of pDP losses of individuals in the dataset induced by different DP algorithms. This is used to provide a theoretically-sound alternative to the prevalent practices of using specific attacks (such as membership inference attacks) for evaluating the data-dependent privacy losses. Before we state the pDP bounds of our algorithms, we extend the standard \((\epsilon(\cdot), \delta)\)-pDP definition to a per-instance version of the Gaussian DP.

Definition 13. We say a mechanism is \( \mu(\cdot) \)-per-instance Gaussian Differentially Private (pGDP), if \( (D, D \cup z) \) (and \( (D \cup z, D) \)) are \((\epsilon, \delta)\)-indistinguishable for all \( \epsilon, \delta \) parameters described by \( \mu(D, z) \).
Algorithm 3: (Theoretical version of) AdaMix training algorithm (no clipping, no adaptive projection, slightly different pretraining, theoretically chosen learning rate).

**Data:** Public dataset $D_{pub}$, private dataset $D_{pri}$, privacy parameter $(\epsilon, \delta)$, noise variance $\sigma$, Lipschitz constant $L^4$, population-level strong convex parameter $c$, regularization parameter $\lambda$ and a constraint set $W$.

**Result:** $\tilde{w}$

// Public Pretraining Phase (OnePassSGD on $D_{pub}$):

$w_1 = 0$. for $t = 1, \ldots, N_{pub}$ do

// In a shuffled order

$\eta_t = \frac{2}{c(t + 1)}$;

$w_{t+1} \leftarrow \Pi_W (w_t - \eta_t \nabla \ell(w_t, (\tilde{x}_t, \tilde{y}_t)))$;

end

$w_{\text{ref}} \leftarrow w_{N_{pub} + 1}$.

// Mix Training Phase (NoisyGD on $D_{pub} \cup D_{pri}$):

$T \leftarrow \text{Calibrate}(\epsilon, \delta, \sigma)$ (i.e., $\frac{T^2 \tau^2}{2 \sigma^2} =: \rho$)

// NoisyGD on objective function $\ell(w) + \frac{\lambda}{2} \| w - w_{\text{ref}} \|^2$

$w_1 = w_{\text{ref}}$;

for $t = 1, \ldots, T$ do

$\eta_t \leftarrow \frac{2}{\lambda(t + T)}$

$n_t \sim N(0, \sigma^2 I)$;

$w_{t+1} \leftarrow \Pi_W (w_t - \eta_t (\sum_{i=1}^{N_{pri}} \nabla \ell_i(w_t) + \sum_{j=1}^{N_{pub}} \nabla \ell_j(w_t) + n_t))$;

end

// The following averaging can be implemented incrementally without saving $w_t$

$\tilde{w} \leftarrow \sum_{t=1}^{T} \frac{2t}{T(T+1)} w_t$

This allows us to obtain precise pDP bounds under composition.

**Proposition 14** (pDP analysis of AdaMix). Let $z_1, \ldots, z_n$ be the data points of the private dataset. Algorithm 3 satisfies $\mu(\cdot)$-pGDP with

$$\mu(D_{-i}, z_i) = \sqrt{\sum_{t=1}^{T} \min\left(\frac{\| \nabla \ell_i(w_t) \|}{\sigma^2}, \tau\right)^2}.$$ 

Similarly, Algorithm 2 satisfies $\mu(\cdot)$-pGDP with

$$\mu(D_{-i}, z_i) = \sqrt{\sum_{t=1}^{T} \frac{\| \sum_{i=1}^{T} \nabla \ell_i(w_t) \|^2}{\tau^2 \sigma^2}}.$$ 

**Proof.** Both algorithms are the composition of $T$-Gaussian mechanisms. Thus the results follow by the composition of pGDP. The composition of pGDP is implied by the composition theorem of GDP (Lemma 8) by choosing the space of datasets to be just $\{D, D \cup \{z\}\}$. \qed

Fixing any $\delta$, we can then compute the corresponding $\epsilon(\cdot)$ for $(\epsilon(\cdot), \delta)$-pDP using the formula of Gaussian mechanism with $\mu$ taken to be $\mu(\cdot)$ pointwise.

**D. Proofs of the technical results**

We will be using the following $O(1/t)$ convergence bound due to Lacoste-Julien, Schmidt and Bach [28], which uses a decaying learning rate and a non-uniform averaging scheme.

---

4 As we discussed earlier, per-example gradient clipping can be viewed as Huberizing the loss function in GLMs [48], all our results apply to the updated loss function. The adaptive clipping approach we took, can be viewed as an heuristic that automatically identifies an appropriate level of Huberization.
Then the (projected) stochastic gradient descent with learning rate \( \eta \) for any fixed \( w \) fresh sample uses a first check that \( \eta \) satisfies that \( \frac{\eta}{m(T+1)} \) is small than \( \lambda B \) where \( \rho \) is the privacy parameter of the algorithm (\( \sqrt{T}p\)-GDP).

**Proof.** First check that \( NL + \lambda B \) upper bounds the Lipschitz constant of \( J \) on because the Lipschitz constant of \( \frac{1}{2} \| w \|^2 \) is smaller than \( \lambda B \) due to the bounded domain. Second, check that the noisy gradient oracle satisfies that it is unbiased, and the added noise has a variance of \( \sigma^2 \) per coordinate for all \( d \) coordinates. Thus

\[
E[\|g_t\|^2 | w_t] = E[\| \nabla J(w_t) \|^2 | w_t] + E[\| n_t \|^2 | w_t] \leq (NL + \lambda B)^2 + d\sigma^2.
\]

Thus by taking expectation on both sides we verify that we can take \( G^2 = (NL + \lambda B)^2 + d\sigma^2 \).

It remains to substitute these quantities and apply the first statement of Theorem 15.

**Corollary 16** (NoisyGD for Strongly Convex Objectives). Let \( J(w) = L(w) + \frac{\lambda}{2} \| w \|^2 \) with individual loss functions \( \ell \) being \( L \)-Lipschitz on \( \mathcal{W} \). Assume \( \sup_{w \in \mathcal{W}} \| w \| \leq B \). Let the learning rate be \( \eta_t = \frac{2}{m(t+1)} \), then

\[
E \left[ J \left( \frac{2}{T(T+1)} \sum_{t=1}^T tw_t \right) \right] - J^* \leq \frac{2(NL + \lambda B)^2}{\lambda T} + \frac{2d\sigma^2}{\lambda T} + \frac{2(NL + \lambda B)^2}{\lambda T} + \frac{dL^2}{\lambda \rho},
\]

where \( \rho := \frac{TL^2}{2\sigma^2} \) is the privacy parameter of the algorithm (\( \sqrt{T}p\)-GDP).

**Proof.** First check that \( NL + \lambda B \) upper bounds the Lipschitz constant of \( J \) on because the Lipschitz constant of \( \frac{1}{2} \| w \|^2 \) is smaller than \( \lambda B \) due to the bounded domain. Second, check that the noisy gradient oracle satisfies that it is unbiased, and the added noise has a variance of \( \sigma^2 \) per coordinate for all \( d \) coordinates. Thus

\[
E[\|g_t\|^2 | w_t] = E[\| \nabla J(w_t) \|^2 | w_t] + E[\| n_t \|^2 | w_t] \leq (NL + \lambda B)^2 + d\sigma^2.
\]

Thus by taking expectation on both sides we verify that we can take \( G^2 = (NL + \lambda B)^2 + d\sigma^2 \).

It remains to substitute these quantities and apply the first statement of Theorem 15.

**Corollary 17** (One-Pass SGD on public data). Assume the public data with \( N \) samples are drawn from the same distribution of the private data. Assume that the (population risk)

\[
R(\theta) = E_{(x,y) \sim D} [\ell(\theta^*, (x, y))]
\]

is \( c \)-strongly convex at \( \theta^* \) for some constant \( c \). Then the one-pass SGD algorithm below

\[
w_{t+1} = w_t - \frac{2}{c(t+1)} \nabla \ell(w_t, (x_t, y_t))
\]

for \( t = 1, ..., N_{pub} \) obeys that

\[
E[\|w_{N_{pub}+1} - w^*\|^2] \leq \frac{4L^2}{c^2 N_{pub}}
\]

where \( w^* = \arg\min_{w \in \mathcal{W}} R(w) \).

**Proof.** First note that since the data is drawn iid, running one-pass SGD by going through the data points in a random order uses a fresh sample to update the parameters. This is equivalent to optimizing the population risk directly. Check that for any fixed \( w \) and all \( t = 1, ..., N_{pub} \), \( E_{(x,y) \sim D} [\nabla w \ell(w, (x, y))] = \nabla R(w) \). Moreover, we need this stochastic gradient oracle to satisfy \( E_{(x,y) \sim D} \| \nabla \ell(\theta, (x, y)) \|^2 \leq G^2 \). Notice that by our assumption \( \| \nabla \ell(\theta, (x, y)) \|^2 \leq L^2 \), thus we can take \( G = L \). By invoking the second statement of Theorem 15 the result follows.

With these two corollaries stated, we are now ready to prove Theorem 3 and Theorem 18.
Proof of Theorem 3. The proof relies on Corollary 16, and the following argument. When additional regularization with parameter $\lambda$, the utility we consider should still be considered in terms of $\mathcal{L}(\hat{w}) - \mathcal{L}(w^*)$. Let $w^*$ be any comparator satisfying $B > \|w^*\|$.

\[
\mathcal{L}(\hat{w}) - \mathcal{L}(w^*) = J(\hat{w}) - J_{\lambda}(w^*) + J_{\lambda}(w^*) - J(w^*) + \frac{\lambda}{2} \|w^* - w_{\text{ref}}\|^2 - \frac{\lambda}{2} \|\hat{w} - w_{\text{ref}}\|^2 \\
\leq J(\hat{w}) - J_{\lambda}(w^*) + \frac{\lambda}{2} \|w^* - w_{\text{ref}}\|^2.
\]

Take expectation on both sides and apply Corollary 16

\[
\mathbb{E}[\mathcal{L}(\hat{w})] - \mathcal{L}(w^*) \leq \frac{2(NL + \lambda B)^2}{\lambda T} + \frac{dL^2}{\lambda \rho} + \frac{\lambda}{2} \|w^* - w_{\text{ref}}\|^2.
\]

Finally, choosing $\lambda = \sqrt{\frac{\rho}{2\|w^* - w_{\text{ref}}\|^2L^2}}$ yields

\[
\mathbb{E}[\mathcal{L}(\hat{w})] - \mathcal{L}(w^*) \leq \frac{4(NL + \lambda B)^2}{\lambda T} + \frac{\sqrt{dL} \|w^* - w_{\text{ref}}\|}{\sqrt{2\rho}}
\]

as claimed (dividing $N$ on both sides to get $\hat{H}$).

\[\square\]

Theorem 18. Assume the private data and public data are drawn i.i.d. from the same distribution $\mathcal{D}$ and that $R(w) = \mathbb{E}_{(x,y)\sim \mathcal{D}}[\ell(w, (x, y))]$ is $c$-strongly convex in $\mathcal{W}$. Let $w_{\text{ref}} = w_{\text{init}} + 1$ — the last iterate of a single pass stochastic gradient descent on $\mathcal{R}_{\text{pub}}(w)$ (initializing from $w_0 = 0$) that goes over the public dataset exactly once one data-point at a time with learning rate $\eta = \frac{2}{c(T+1)}$. Let $w_{\text{ref}}$ be passed into Theorem 3’s instantiation of NoisyGD, which returns $\hat{w}$ (The pseudo-code of this algorithm is summarized in Algorithm 3 in the appendix), then at the limit when $T$ is sufficiently large, the excess risk obeys that

\[
\mathbb{E}[R(\hat{w})] - R(w^*) \\
\leq \frac{4\sqrt{dL^2}}{c \sqrt{N_{\text{pub}}} N \sqrt{2\rho}} + \text{Gen}(\hat{w}, N) + \text{Gen}(w^*, N),
\]

where $w^* = \arg\min_{w\in\mathcal{W}} R(w)$ and $\text{Gen}(w, N) := \left| \mathbb{E}[R(w)] - \hat{R}(w) \right|$ is the expected generalization gap of (a potentially data-dependent) $w$.

Proof of Theorem 18. Let $w^* = \arg\min_{w\in\mathcal{W}} R(w)$. By Corollary 17, we have

\[
\mathbb{E}[\|w_{\text{ref}} - w^*\|^2] \leq \frac{4L^2}{c^2 N_{\text{pub}}},
\]

which implies (by Jensen’s inequality) that $\mathbb{E}[\|w_{\text{ref}} - w^*\|] \leq \sqrt{\frac{4L^2}{c^2 N_{\text{pub}}}}$.

Now by plugging in the $w^*$ in theorem, take expectation over the public dataset, and substitute the above bound, we get

\[
\mathbb{E}[\mathbb{E}[\hat{R}(\hat{w})] - \hat{R}(w^*)] \leq \frac{4(NL + \lambda B)^2}{\lambda T N} + \frac{2\sqrt{dL^2}}{c \sqrt{N_{\text{pub}}} \sqrt{2\rho}}.
\]

Take $T$ to be sufficiently large so that the second term dominates, we obtain the stated bound.

Finally, to convert the above bound into that of the excess risk:

\[
\mathbb{E}[\mathbb{E}[\hat{R}(\hat{w})] - \hat{R}(w^*)] - \mathbb{E}[R(\hat{w})] - R(w^*) \\
\leq \mathbb{E}[\mathbb{E}[\hat{R}(\hat{w})] - \hat{R}(w^*)] + |\mathbb{E}[\hat{R}(w^*)] - R(w^*)| \\
:= \text{Gen}(\hat{w}, N) + \text{Gen}(w^*, N),
\]

which completes the proof. 

\[\square\]
We make two additional remarks. First, we do not require the empirical objective $L_{\text{pub}}$ to be strongly convex. In practice, we do not have strong convexity when $N_{\text{pub}} < d$. The assumption of $c$-strong convexity is on the population-level risk function $R$. Second, our bound decomposes the excess (population) risk of the private learner into a (local) uniform-convergence bound (which is required by a non-private learner too) and an additional cost due to privacy. Note that Gen$(N)$ is usually $O(1/\sqrt{N})$ but could be $O(1/N)$ when certain data-dependent “fast rate” conditions are met, e.g., realizability, low-noise, or curvature (see, e.g., [27]). Our results suggest that the cost of privacy asymptotically vanishes (fix $\rho$, $N_{\text{pub}} \to \infty$, and $N_{\text{pub}}/N \to 0$) even under these fast rate conditions relative to the non-private rate.