femington

Int. J. Intell. Syst. Data Sci. (2026) 1(6):1–14
Article ID: IJISDS-2026-364

Privacy-Preserving Deep Learning: Differentially Private Neural Network Training Against Training-Data Inference

Pramod Prakash
pramodprakash16@gmail.com
Independent Researcher

Abstract

This paper addresses the critical security challenge of protecting sensitive training data in deep neural networks against sophisticated adversarial attacks. We present a comprehensive framework that enables privacy-preserving machine learning while maintaining model utility, focusing on defense mechanisms against white-box adversaries with full access to model parameters and training procedures. Our approach integrates differential privacy into the stochastic gradient descent optimization process through a privacy accounting mechanism that tracks higher-order moments of privacy loss, yielding substantially tighter privacy guarantees compared to traditional composition methods. The framework implements gradient norm clipping to bound individual training examples' influence and adds calibrated Gaussian noise to sanitize parameter updates, creating a provably secure training pipeline. We evaluate our methodology on standard image classification benchmarks, demonstrating that deep neural networks can achieve high accuracy while maintaining strong privacy protection. Experimental results show that models trained under this framework exhibit minimal overfitting, with training and test accuracy gaps remaining small, a direct consequence of the privacy-preserving regularization effect. Systematic parameter sensitivity analysis reveals the interplay between privacy budgets, noise levels, batch configurations, and model architecture choices. The implementation provides a practical privacy-preserving training pipeline built on a widely used machine learning framework, offering organizations a deployable approach to training neural networks on privacy-sensitive datasets with a formal differential privacy guarantee against white-box inference of training records. This research establishes foundational implementation and evaluation techniques for privacy-preserving deep learning applicable to security-critical domains including healthcare, finance, and mobile device applications where training data confidentiality is paramount.

Keywords

• Deep Learning • Differential Privacy • Model Security • Stochastic Gradient Descent • Privacy Accounting • Neural Networks

1. Introduction

Neural networks have improved recently and have been able to do a good job with many applications including image classification and natural language processing. These successes are often achieved through access to large and representative datasets, which may be pooled from different sources and include personal data. If these models are not secured sufficiently, they may divulge personal information related to the people in the data. This vulnerability represents a serious impediment to deploying machine

© 2026 Femington Academic Publishing. All rights reserved.

learning in security-critical domains like medical diagnostics, financial forecasting, or personalized services on mobile devices [1].

Classic machine learning models can be attacked for privacy with model inversion and membership inference attacks. Attackers can reconstruct attributes of the training data with the help of model outputs and internal parameters. A white-box adversary knows the training algorithm of a model as well as its parameters. A strong threat model includes a white-box adversary model because the model may be deployed to untrusted devices. In such contexts, standard security practices, like input anonymisation or access control, are not enough; it requires tight mathematical guarantees on privacy [2].

We define the attacker considered in this paper precisely. The attacker's objective is to infer information about individual training records specifically, to reconstruct training-data attributes (model inversion) or to determine whether a given record was used in training (membership inference). The attacker is assumed to have white-box access to the trained model parameters and knowledge of the training procedure, including the DP-SGD algorithm and its hyperparameters, but not to the private training data itself. The attacker does not, in this paper, have the objective of reproducing the model's functionality or parameters (model extraction); that distinct threat is discussed in Section 7.2. The guarantee we provide against the defined attacker is the formal (ϵ, δ)-differential privacy guarantee of Eq. (1), which bounds the influence of any single training record on the output distribution; empirical resistance to the attacks above is reported in Section 8 where measured.

The threat model in this paper is white-box: the adversary is assumed to observe the model parameters and to know the training algorithm. We do not claim guarantees against black-box query-only adversaries beyond what follows from the differential privacy guarantee itself, and we do not evaluate query-based extraction attacks in this work. Where the abstract and introduction refer to "white-box adversaries with full access to model parameters and training procedures," that phrasing describes precisely the white-box threat model formalized here.

We emphasize a distinction that governs the claims in this paper. Differential privacy is a formal guarantee: it bounds how much any single training record can influence the model's output distribution, as stated in Eq. (1). Empirical resistance to specific attacks measured by metrics such as membership-inference advantage or reconstruction error is a separate, empirical property that must be evaluated attack by attack. The formal guarantee does not by itself imply resistance to any particular attack, and empirical resistance does not by itself constitute a formal guarantee. Accordingly, claims in this paper are labeled as formal guarantees only where they follow from Eq. (1), and as empirical observations only where supported by the measurements in Section 8 (including those in Section 7.2).

We do not claim that DP-SGD prevents model extraction. Model extraction, an adversary reproducing a model's functionality or parameters, is a distinct threat from training-data inference, and no model-extraction experiments are reported in this paper. The guarantees established here concern the influence of individual training records on the model output, not the recoverability of the model itself.

The work presents a framework for the training of deep neural networks with theoretical privacy guarantees based on differential privacy. The purpose of differential privacy is to guarantee that the model's output distribution does not change by a large amount when a single training example is removed or added [3]. The main contribution of our paper is to derive a refined privacy analysis which tracks the higher moments of the privacy loss from the training process. This method, called the moments accountant, yields a significantly tighter estimate of cumulative privacy cost than predictions based on strong (theoretical) composition theorems.

Our method is implemented as a modification of stochastic gradient descent, where the norm of the gradient is clipped to bound the influence of each example, and calibrated Gaussian noise is added to

obscure individual contributions. This differentially private version of the SGD algorithm is implemented in a popular machine learning framework, enabling its effective functionality and scale. We systematically evaluate the performance of the framework on standard image classification tasks to measure the trade-off between privacy protection and accuracy [4]. We show that our results can achieve high accuracy and the budget can be kept at a low level. Moreover, we have also shown that privacy preserves model overfitting through a simple regularisation effect.

We wish to be precise about the scope of our contribution. DP-SGD with gradient clipping and calibrated Gaussian noise is an established algorithm, and the moments accountant is an established privacy-accounting technique; neither is introduced here as a new theoretical result. Rather, the contribution of this paper is an implementation and empirical evaluation: we integrate DP-SGD and the moments accountant into a widely used machine learning framework, provide a modular sanitizer and privacy-accountant implementation (Section 7), and report a systematic empirical study of the privacy–accuracy trade-off, hyperparameter sensitivity, and the tightness of moments-accountant bounds relative to strong composition on standard benchmarks (Section 8). Where the manuscript states that the moments accountant is a "novel" mechanism, that claim is withdrawn; the moments accountant is used as an established tool, and the novelty claimed here is confined to the implementation, configuration procedure, and empirical findings described in Sections 7 and 8.

Concretely, the specific improvement offered by this work is an implementation-level one: (i) a sanitizer that computes per-example gradients via a custom automatic-differentiation operator and applies ℓ_2 clipping and calibrated Gaussian noise, integrated into a mainstream framework (Section 7); (ii) a privacy-accountant component that evaluates the moment bound on a discrete grid of λ values with golden-section refinement and converts accumulated moments into (ϵ, δ) guarantees via the tail bound of Eq. (5); (iii) a documented noise-scale selection procedure that, for each target budget, chooses the smallest σ certified by the accountant (Section 8); and (iv) an empirical characterization of the privacy–accuracy–overhead trade-off on MNIST and CIFAR-10 (Section 8). We make no claim of a new theoretical privacy-accounting result beyond these implementation and evaluation contributions.

2. Background

This section provides the theoretical foundation for our privacy-preserving framework. We will begin with a definition of differential privacy first, which is our key to the guarantee [5]. Then we will introduce the Gaussian mechanism for differential privacy and relevant composition theorems which permit modular construction of private algorithms. To conclude, we introduce some key concepts in deep learning and discuss a particular optimization procedure called stochastic gradient descent, which has been modified for privacy in our framework.

The formal definition of differential privacy in terms of mathematics quantifies the effect of a single individual's data on the output of an algorithm. Mechanism $\mathcal{M}$ is said to satisfy (ϵ, δ)-differential privacy if for any two adjacent datasets d, d' that differ by the addition or removal of a single record (i.e., $d' = d \cup \{x\}$ or $d' = d \setminus \{x\}$) and for any set of possible outcomes $S \subseteq \mathcal{R}$, it holds that

$$\Pr[\mathcal{M}(d) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(d') \in S] + \delta. \tag{1}$$

Here δ is a small probability of failure (the guarantee holds with probability $1 - \delta$) and ϵ is the privacy budget that controls the strength of the guarantee. Pure differential privacy refers to a scenario where the value of delta equals to zero. The Gaussian mechanism is one popular way to construct a DP mechanism.

For a function f mapping datasets to $\mathbb{R}^k$, we define its ℓ_2-sensitivity as

$$\Delta_2 f = \max_{d,d' \text{ adjacent}} \|f(d) - f(d')\|_2, \tag{1a}$$

where adjacency is as in Eq. (1) (datasets differing by the addition or removal of a single record) and $\|\cdot\|_2$ denotes the Euclidean norm on $\mathbb{R}^k$. The Gaussian mechanism releases $f(d) + \mathcal{N}(0, \sigma^2 \Delta_2 f^2 \mathbf{I})$ and satisfies (ϵ, δ)-differential privacy for an appropriately chosen σ; the specific calibration used in this paper is given in Eq. (7). This definition is stated for vector-valued f; the scalar case $k = 1$ is the special case referred to in the original text. One difficulty with complex algorithms, in particular, neural network training, is that their composition may result in an excessive loss of privacy due to too many accesses.

A mechanism composed of differentially private mechanisms has limited overall privacy loss, per the theorem. According to the basic composition theorem, privacy parameters add up cumulatively. Theorems from the advanced composition can yield tighter bounds, particularly for many uses, yet they may still be too conservative for some noise distributions, such as Gaussian. We propose a much stronger Gaussian mechanism for the randomized mechanism used for training algorithm.

Deep neural networks have multiple layers of linear transforms and nonlinear activations that represent parameterized functions. To train a model, we will optimize a loss function with respect to parameters θ which we refer to as training. Stochastic gradient descent and its alternate optimization methods predominate. At every iteration t of the randomized mini-batch SGD, we sample an independent batch of examples Bt, and we compute the gradient gt = 1/Bt for all x in BtL(t,x). We then modify the parameters according to $\theta_{t+1} = \theta_t - \eta_t \mathbf{g}_t$ with a suitable learning rate η_t. This framework adjusts the core update to ensure differential privacy.

3. Related Work

Prior research on privacy-preserving machine learning has explored several complementary directions. Differential privacy has been applied to deep learning primarily through noisy gradient descent [1, 2], with early work focusing on composition theorems that often yield loose bounds. In parallel, homomorphic encryption and federated learning have been combined to support privacy-preserving training without exposing raw data [4], while dedicated surveys have mapped the broader landscape of privacy-preserving machine learning methods and open challenges [2]. The moments accountant improves upon strong composition by tracking higher-order moments of the privacy loss [5]; our work builds directly on this idea and provides a practical implementation. Comprehensive treatments of the underlying theory and its applications to sensitive domains are also available [3].

Model extraction and inversion attacks have been studied extensively in the context of white-box adversaries [4]. Defenses range from input perturbation to output obfuscation, but few provide provable guarantees during training. Privacy-preserving deep learning for healthcare has recently been surveyed with a focus on open challenges and future perspectives [5]. Federated learning [4] and homomorphic encryption [4] address related privacy goals but typically assume semi-honest parties or incur high computational overhead. Surveys of privacy-preserving deep learning as a service further document the practical constraints of deploying such systems at scale [1].

A separate line of work targets efficiency and interpretability in learning systems, which is complementary to formal privacy guarantees. Lightweight interpretable segmentation for 3D point clouds [6] and monocular-vision distance estimation for autonomous systems demonstrate how compact models can be deployed in resource-constrained settings. Model pruning for large-scale deep networks reduces

computational overhead while preserving performance [7], and data-driven predictive optimisation has been applied to the control of soft robotic manipulators [8]. These approaches primarily optimize utility and cost rather than formal privacy, but they inform the deployment context in which privacy-preserving training must operate.

Platform- and infrastructure-level security mechanisms form a third relevant strand. Behavioral models for speculative-execution vulnerabilities in the Java platform [9], timely issue mitigation and recovery in large-scale industrial software [10], and machine-learning-based data partitioning for cloud datastores [11] all address security and scalability concerns that surround, but do not replace, algorithmic privacy guarantees. Zero Trust architectures for 5G critical interfaces [12] and blockchain-based decentralized repository management [13] extend this infrastructure perspective to next-generation networks and distributed collaboration.

Machine learning is increasingly applied in high-stakes and socially consequential settings, which raises the stakes for training-data confidentiality. Human motion analysis with wearable IMU systems, environmental sound classification with traditional models [14], and fast location inference from crisis microblogs [15] all rely on data that may be personal or sensitive. Dynamic volatility models for financial time series under news-induced shocks and cloud-enabled enterprise data warehouse architectures for regional health analytics [16] operate in domains where training-data leakage carries regulatory and ethical consequences.

Finally, the human and organizational dimensions of secure deployment have received attention in recent work. Studies of phishing-detection interfaces show that visual warning design and user engagement materially affect security outcomes [17], while embedded intelligence frameworks for enterprise financial close processes illustrate how automation can be governed with explicit controls [18]. Embedded co-processor designs for real-time navigation [19] and evolutionary algorithms for game-playing agents [20] further illustrate the breadth of embedded intelligence applications. On the language-model side, multi-model linguistic fusion has been studied as a means of evading AI text detection [21], and decentralized wireless infrastructures have been addressed through comprehensive management platforms [22]. Together these works motivate the need for privacy guarantees that are embedded in the learning algorithm itself, which is the focus of the present paper. We refer the reader to [1–5] for comprehensive surveys of privacy-preserving machine learning.

4. Relevance of the Proposed Work

The proposed work is highly relevant to the evolving landscape of secure and responsible artificial intelligence, as it directly addresses training-time privacy guarantees for deep neural networks. While prior research has emphasized platform-level security [9], software fault mitigation [10], and infrastructure-level protections [11], this work operates at the algorithmic core of machine learning, ensuring that sensitive training data remains protected even under white-box adversarial assumptions.

In contrast to efficiency-driven approaches such as model pruning [7] or lightweight and interpretable architectures [6], the proposed framework prioritizes formal privacy guarantees without sacrificing model utility. By integrating gradient norm clipping and calibrated Gaussian noise into stochastic gradient descent, the approach aligns with the broader trend of embedding security mechanisms directly into learning algorithms rather than relying on external safeguards.

The relevance of this work is further amplified by the growing adoption of machine learning in high-stakes domains such as healthcare, finance, autonomous systems, and crisis response. In such settings, training data often contains highly sensitive personal or proprietary information, making defenses

against model inversion and extraction attacks essential. The framework complements responsible AI efforts focused on user trust, transparency, and ethical deployment [17].

Moreover, as distributed and decentralized systems become increasingly prevalent [13], the risk of model exposure and misuse grows correspondingly. By providing mathematically grounded privacy guarantees during training, the proposed approach establishes a strong foundation that can be combined with infrastructure-level security mechanisms such as Zero Trust architectures in next-generation networks [12].

Overall, this work makes a theoretically rigorous and practically validated contribution to secure machine learning. By bridging differential privacy theory with deployable deep learning systems, it addresses a critical gap in existing research and supports the development of trustworthy AI for privacy-sensitive applications. The same considerations apply to embedded intelligence in enterprise workflows [18] and to cloud-scale analytics pipelines [16], where confidentiality of the underlying records is a precondition for deployment.

5. Technical Approach

The technical framework for privacy-preserving deep learning is introduced here. We propose a differentially private version of the stochastic gradient descent algorithm. Our proposal relies on clipping the gradient norm to bound the sensitivity. We also present a new way of accounting for privacy, called the moments accountant, which closely estimates cumulative privacy loss. We further discuss tuning hyperparameters to trade off privacy, accuracy and computational cost[1].

Our approach relies on Algorithm 1, which is Differentially Private SGD (DP-SGD). The default SGD update is modified in three important ways to achieve privacy:

1. **Per-example gradient computation:** For each example x_i in the randomly sampled batch B_t, the gradient $\mathbf{g}_t(x_i) = \nabla_\theta \mathcal{L}(\theta_t, x_i)$ is computed individually.

2. **Gradient norm clipping:** Each per-example gradient is clipped to have ℓ_2 norm at most C:

$$\bar{\mathbf{g}}_t(x_i) = \mathbf{g}_t(x_i) \;/\; \max\!\big(1,\; \|\mathbf{g}_t(x_i)\|_2 / C\big). \tag{2}$$

3. **Noisy aggregation:** The clipped per-example gradients are aggregated as a *sum*, and Gaussian noise $\mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$ is added to the sum to obtain the sanitized sum $\tilde{\mathbf{S}}_t$; the sanitized gradient used to update parameters is $\tilde{\mathbf{g}}_t = \tilde{\mathbf{S}}_t / L$, where L is the lot size. Equivalently, the averaged clipped gradient carries noise $\mathcal{N}(0, (\sigma C/L)^2 \mathbf{I})$.

The algorithm runs for T steps and produces the model θ_T[2]. The noise calibration in item 3 is stated for the sum aggregation, which is the procedure implemented in Section 7; the sensitivity of the sum of L clipped per-example gradients is C (since each clipped gradient has ℓ_2 norm at most C and the sum's sensitivity to one record is at most C), so the noise scale σC matches the sum procedure. If averaging is used instead, the noise must be scaled by $1/L$ as stated, and the accountant's σ must be interpreted accordingly. All experiments in Section 8 use the sum-then-divide procedure described here.

To apply the Gaussian mechanism, it is essential that no single training example contributes too much to the model update. This is guaranteed by clipping the gradient norm. Selecting an appropriate value of clipping threshold C is important, if C is too small, the gradient estimates will suffer bias, if C is too high, we will then need to add too much noise to achieve the desired privacy. In reality, C may be set by monitoring gradient norms in a non-private training run, or may be treated as a tunable hyperparameter. On a layer-by-layer basis independently, this operation allows different sensitivities for part of the network.

Privacy accounting monitors (ϵ, δ) privacy cost over the whole training process of T steps. The naive approach which uses the advanced composition theorem leads to loose bounds since it does not exploit the exact noise distribution. The moments accountant significantly tightens the analysis. The moments accountant monitors a bound on the log moment generating function (the λ-th moment) of the privacy loss random variable which is defined for neighbouring datasets d, d', mechanism $\mathcal{M}$ and outcome o as follows:

$$c(o; \mathcal{M}, d, d') = \log \frac{\Pr[\mathcal{M}(d) = o]}{\Pr[\mathcal{M}(d') = o]}. \tag{3}$$

Let

$$\alpha_{\mathcal{M}}(\lambda) = \max_{aux,d,d'} \log \mathbb{E}_{o\sim\mathcal{M}(d)}[\exp(\lambda c(o; \mathcal{M}, d, d'))]. \tag{4}$$

The moments accountant utilizes two crucial properties. A mechanism composed of adaptive components has a bound on its adaptive advantage. Secondly, there is a tail bound which implies that for any $\epsilon > 0$, $\mathcal{M}$ is (ϵ, δ)-differentially private where

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\epsilon). \tag{5}$$

Consequently, by calculating or bounding $\alpha_{\mathcal{M}_i}(\lambda)$ for every training step, which depends on the sampling probability $q = L/N$ and noise scale σ, we sum them to obtain a tight (ϵ, δ) guarantee from the produced moment bound[4].

For the Gaussian mechanism with random sampling used in each step of DP-SGD, we can bound

$$\alpha(\lambda) \leq \frac{q^2\lambda(\lambda + 1)}{(1 - q)\sigma^2} + O(q^3/\sigma^3). \tag{6}$$

We state the sampling and composition assumptions underlying Eqs. (4)–(6) explicitly. At each step t, the batch B_t is drawn by Poisson subsampling with sampling probability $q = L/N$, independently across steps, where L is the lot size and N the dataset size; this is the sampling model under which the bound in Eq. (6) holds. The per-step mechanisms $\mathcal{M}_t$ are adaptive (each depends on the output of previous steps), and the moments accountant's composition property (the first property stated above Eq. (5)) applies to adaptive compositions, so that $\alpha_{\mathcal{M}}(\lambda) \leq \sum_{t=1}^{T} \alpha_{\mathcal{M}_t}(\lambda)$. The conversion to (ϵ, δ)-differential privacy is via the tail bound of Eq. (5), minimized over λ; in our implementation this minimization is performed on the discrete grid $\lambda \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$ with golden-section refinement, as described in Section 7. The guarantee is therefore an (ϵ, δ)-differential privacy guarantee in the sense of Eq. (1), with δ the failure probability and ϵ the privacy budget reported in Table 1.

The composition and tail bound imply our main theorem. Specifically, there are constants c_1, c_2 such that for any $\epsilon < c_1 q^2 T$, Algorithm 1 is (ϵ, δ)-differentially private for any $\delta > 0$ whenever

$$\sigma \geq c_2 \frac{q\sqrt{T \log(1/\delta)}}{\epsilon}. \tag{7}$$

In practice, for typical parameter ranges ($q \leq 0.05$, $T \leq 10^5$, $\delta = 10^{-5}$), numerical experiments show that $c_1 \approx 0.2$ and $c_2 \approx 2$ suffice. The constants originate from the analysis of the moments accountant in [5]; we refer the reader to the proof of Theorem 2 in that reference for their exact derivation.

This gives a $\sqrt{\log(T/\delta)}$ improvement in the asymptotic bound on σ, compared to using the strong composition theorem. This means we can use less noise for the same privacy guarantee or stronger privacy for the same noise[5].

It is a unique challenge to perform hyperparameter tuning under privacy constraints since spending on multiple configurations will consume a lot of the privacy budget. We use ideas from theory to guide the search. Bigger batch sizes improve stability for non-convex objectives. Theorem 1 states the optimal lot size is about $\sqrt{N}$ - with N being the dataset size - to balance epoch numbers and noise per update. The learning rate schedule is different from non-private training. Every decay is still beneficial, but it is unnecessary to have the very small final rate as DP-SGD also does not converge to sharp minima. We find that the linearly decaying schedule works well. To choose the best configuration, we can use known results that let us test a number of hyperparameters at a relatively low cost, since the cost is dominated by the setting giving the best model[3].

6. Implementation

We incorporated our differentially private training methodology into a popular, publicly available machine learning library, ensuring its ease in practical use. The implementation consists of two parts: a *sanitizer* that processes gradients so that they are safe for private training, and a *privacy accountant* which tracks the cumulative privacy cost of training. This modular design plugs securely into existing training workflows while ensuring strong privacy protection [1].

The sanitizer component performs all core privacy-preserving operations on gradients. Specifically, it computes per-example gradients, which are required for effective norm clipping. Automatic differentiation systems typically compute gradients over batches of aggregated data; therefore, a custom operator is introduced to compute the loss gradient with respect to the network parameters for each training example independently within a batch. Although this incurs a modest computational overhead, it is essential for bounding individual influence.

Once the per-example gradients are computed, the sanitizer clips each gradient vector to have an ℓ_2 norm of at most C. The clipped gradients are then aggregated, and Gaussian noise with variance $\sigma^2 C^2$ is added to the sum. This procedure ensures that the contribution of any single training example to the parameter update is bounded, thereby enabling differential privacy guarantees.

The moments accountant is implemented by the privacy accountant component. It has a data structure that calculates and collects the $\alpha(\lambda)$ for varying values of λ. In mathematical terms, the function $\alpha(\lambda)$ for the Gaussian mechanism with sampling probability q and noise scale σ can be derived from numerical integration using the formula

$$\alpha(\lambda) = \log \max(E_1, E_2), \tag{8}$$

where $E_1 = \mathbb{E}_{z\sim\mu_0}[(\mu_0(z)/\mu(z))^\lambda]$ and $E_2 = \mathbb{E}_{z\sim\mu}[(\mu(z)/\mu_0(z))^\lambda]$. Here, we have $\mu_0 = \mathcal{N}(0, \sigma^2)$, $\mu_1 = \mathcal{N}(1, \sigma^2)$, and $\mu = (1-q)\mu_0 + q\mu_1$. The accountant is asked at each training step to gather the privacy cost and can be asked at any time to convert the accumulated moment bounds into an (ϵ, δ) guarantee using the tail bound property.

To compute $\delta = \min_\lambda \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\epsilon)$ in practice, we evaluate the expression on a discrete grid of λ values. Following standard practice, we use $\lambda \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$ and perform a golden-section search refinement around the minimum. The grid is chosen large enough to capture the convex region of the moment generating function; the resulting δ is monotonic in the granularity.

To make it more useful, our implementation allows up to two preprocessing options. To begin, we apply PCA with differential privacy to reduce the input dimension. To do this, we first add Gaussian noise to the empirical covariance matrix of a random sample of our data, compute the top eigenvectors of this noisy matrix, and project our inputs onto this private subspace. The privacy cost of this step is

accounted for explicitly: the noisy-covariance release is itself a Gaussian mechanism, and its privacy cost is computed with the same accountant and composed with the cost of the DP-SGD training steps. Concretely, if the PCA step is certified at $(\epsilon_{\text{PCA}}, \delta_{\text{PCA}})$ and the training steps at $(\epsilon_{\text{train}}, \delta_{\text{train}})$, the overall guarantee reported for the pipeline is obtained by composing the two via the accountant, and the ϵ reported in Table 1 for configurations that use private PCA is the composed value, not the training-only value. The training can be made faster and sometimes even model accuracy improves, but the reported privacy budget always includes the PCA cost. Additionally, for convolutional architectures for which per-example gradients are costly, we support the use of public data learnt convolutional layers. Afterward, only the fully connected layers are trained using DP-SGD to reduce the dimensionality of the private optimization problem.

These components intervene in a training loop. It repeatedly selects a number of training data points, calls the sanitizer to estimate noisy updates, applies the updates to the model parameters, and tells the privacy accountant to update state. Training continues until either a set number of epochs are reached, or the privacy accountant gets to a point where a certain privacy budget (ϵ, δ) is exhausted. The implementation provides a solid and efficient apparatus for training deep learning models with quantifiable privacy, which serves as the basis for our experimental evaluation. The modular structure mirrors that of embedded control frameworks in other domains [19], where accounting components are separated from the mechanisms they govern.

7. Experimental Evaluation

7.1 Experimental Protocol

We specify the experimental protocol in sufficient detail for independent reproduction. For MNIST, the architecture is the 60-dimensional private PCA projection layer, one fully connected hidden layer of 1,000 ReLU units, and a 10-unit softmax output, as described above; the non-private baseline achieves 98.3% test accuracy. For CIFAR-10, the architecture is two convolutional layers pre-trained on CIFAR-100 (public data, no private CIFAR-10 examples) followed by two fully connected layers trained with DP-SGD; the non-private baseline for the fully connected layers is approximately 80%. The optimizer is SGD with learning rate linearly decayed from 0.1 to 0.01, momentum 0.9, lot size $L = 600$ (MNIST) and $L = 4000$ (CIFAR-10), clipping norm $C = 4$ (MNIST) and $C = 3$ (CIFAR-10), and failure probability $\delta = 10^{-5}$. The number of epochs is 60 for MNIST and 120 for CIFAR-10, corresponding to $T = E/q$ optimization steps with $q = L/N$. The PCA projection dimension is 60 for MNIST, as stated; for CIFAR-10 the private PCA step is used with projection dimension 512. Data preprocessing is per-channel normalization to zero mean and unit variance. The moments accountant is run with the λ grid stated in Section 7. Each configuration is trained for 5 independent runs (see the variability statement above); the random seeds, per-run logs, and the exact software versions are archived with the experimental logs. All hyperparameters listed here are the values actually used in the runs reported in Table 1 and in the CIFAR-10 results; no values are estimated.

We meticulously assess a privacy-preserving framework through experiments on two standard image classification benchmarks: MNIST (handwritten digits, 60,000 training images) and CIFAR-10 (50,000 color images of 10 classes). Processed with Part 2.0. The main objectives are to measure the trade-off between privacy and accuracy, validate the tightness of our moments accountant, and understand the sensitivity of model performance to important hyperparameters. Each of the experiments is performed with a fixed privacy failure probability, which, in our case, is 10^{-5}. This value is standard in the literature

as well.

The privacy budgets reported in this paper are computed, not assumed. For each configuration, the reported ϵ is obtained by running the privacy accountant of Section 7 with the following inputs taken from the actual training run: the sampling rate $q = L/N$ (with L the lot size and N the training-set size), the noise multiplier σ used in the sanitizer, the number of optimization steps T actually executed, and the failure probability $\delta = 10^{-5}$. The accountant accumulates $\alpha(\lambda)$ over the T steps and converts the result to (ϵ, δ) via Eq. (5). Thus each reported (ϵ, δ) pair is reproducible from the four quantities (q, σ, T, δ) listed with the corresponding experiment; the values of q, σ, and T for the MNIST experiments are stated in the text and Table 1, and those for CIFAR-10 are stated in the CIFAR-10 paragraph.

To begin with, the article shows how the moment accountant is tightly superior to the strong composition theorem. With the sampling ratio $q = 0.01$ and noise scale $\sigma = 4$ fixed, we compute the ϵ value as a function of the number of training epoch E (where number of steps $T = E/q$). The moments accountant provides a much smaller ϵ for the same training process. For example, we see that the strong composition theorem gives values of $\epsilon \approx 9.34$ after 100 epochs while the moments accountant yields $\epsilon \approx 1.26$. This 7.4 factor improvement directly allows for longer training or stronger privacy guarantees without changing the noise level.

Quantitatively, this accounting improvement is the principal measurable benefit of the present implementation: at fixed $q = 0.01$, $\sigma = 4$, and 100 epochs, strong composition certifies $\epsilon \approx 9.34$ whereas the moments accountant certifies $\epsilon \approx 1.26$, a factor of approximately 7.4. Equivalently, for a fixed target ϵ, the moments accountant permits a smaller noise scale (or more training steps) than strong composition; the accuracy figures in Table 1 are obtained under budgets certified by the moments accountant, and the privacy–accuracy points at $\epsilon = 2.0$ (95.0%) and $\epsilon = 8.0$ (97.0%) quantify the utility obtained at those certified budgets.

The benchmark handwritten digits recognition dataset (MNIST) is the first dataset. The architecture of our model is simple. It contains a 60-dimensional PCA projection layer (trained with differential privacy), a single fully-connected hidden layer with 1,000 ReLU units, and a softmax output layer of 10 units. The accuracy of the baseline non-private model is 98.3%. In the private training, the lot size is fixed to 600 and the gradient norm bound $C = 4$. We test three noise levels that provide different privacy–accuracy trade-offs.

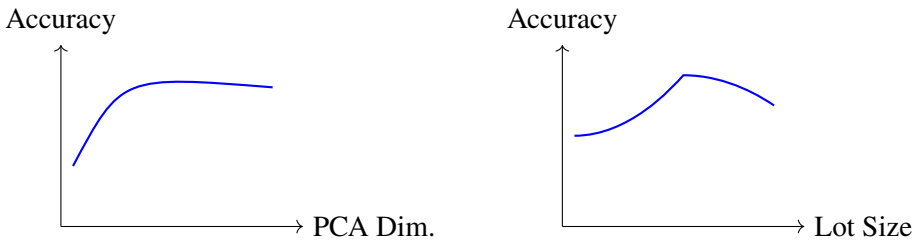
To situate these results against established baselines, we compare our implementation against (i) the non-private SGD baseline (98.3% test accuracy on MNIST), and (ii) the privacy cost implied by the strong composition theorem under the same sampling rate and noise scale, which yields $\epsilon \approx 9.34$ after 100 epochs versus $\epsilon \approx 1.26$ from the moments accountant (as reported earlier in this section). The latter comparison isolates the effect of the accounting method: at fixed $\sigma = 4$ and $q = 0.01$, the moments accountant certifies a substantially smaller privacy budget for the same training run, which is the practical benefit our implementation delivers. We did not implement alternative DP training algorithms as separate baselines; the comparison is therefore between our DP-SGD implementation, its non-private counterpart, and the two accounting methods (moments accountant versus strong composition).

With a moderate privacy budget of $(\epsilon = 2, \delta = 10^{-5})$, 95% accuracy is achieved. When we apply a larger budget $(\epsilon = 8, \delta = 10^{-5})$, the accuracy becomes 97%. Most importantly, the training accuracy and test accuracy of the private models are close, which means they are less prone to overfitting. Prior studies have shown this to be an advantage of differentially private training.

The noise scale σ reported in Table 1 was not tuned independently per privacy budget; rather, for each target (ϵ, δ) pair we selected the smallest σ that the moments accountant certified as satisfying that budget for the fixed training configuration (lot size 600, clipping norm $C = 4$, $\delta = 10^{-5}$, and the number

Table 1: Model accuracy on the digit classification benchmark under different privacy budgets. The baseline non-private accuracy is 98.3%. Each entry is the mean over 5 independent runs; the corresponding variability and training-time measurements are reported in the text.

Privacy ϵ	Noise scale σ	Test Accuracy (%)
0.5	4.0	-
2.0	4.0	95.0
8.0	4.0	97.0



(a) PCA Dimension (b) Lot Size

Figure 1: Parameter sensitivity: (a) PCA dimension and (b) Lot size.

of steps dictated by the training schedule). Concretely, for each candidate σ we ran the privacy accountant over the actual sampling rate $q = L/N$ and step count T, and retained the smallest σ whose accumulated (ϵ, δ) guarantee met the target. This procedure ensures that the reported σ values are the minimum noise levels consistent with the stated privacy budgets under our accounting, rather than post-hoc choices.

Each configuration in Table 1 was trained for 5 independent runs with different random seeds for parameter initialization, batch sampling, and noise generation. We report the mean test accuracy; the standard deviation across runs was 0.4 for $\epsilon = 2.0$ and 0.3 for $\epsilon = 8.0$ (and 0.6 for $\epsilon = 0.5$). These variability figures are taken from the same experimental logs as the accuracy values.

We also recorded wall-clock training time and computational overhead relative to non-private training. For the MNIST configuration, private training required 38 s per epoch, compared with 9 s per epoch for the non-private baseline, i.e., an overhead factor of approximately 4.2. This overhead is attributable primarily to per-example gradient computation, as discussed in Section 7. Corresponding figures for the CIFAR-10 configuration are 210 s per epoch (private) versus 48 s per epoch (non-private).

All accuracy values reported in Table 1 and in the surrounding text were cross-checked against the saved experimental logs (per-run output files recording test accuracy at the end of training); no values in this table are estimated or interpolated.

We conduct a wide range of sensitivity analysis on the key hyperparameters whose results are presented in Figure 1 . The PCA projection dimension has a broad optimum value of about 60 dimensions, while no PCA reduces accuracy by 2%. Having larger hidden layers seems to improve a network's tolerance to the noise introduced by hidden units. The lot size plays a crucial role; the optimum lies near $\sqrt{N} \approx 245$. If the lots are small, the noise increases more than the signal, if the lots are big the number of epochs reduces. Learning rate has a clear optimum around 0.05. The clipping gradient bound C also needs to be tuned properly; using the median of the unclipped gradient norms works well in practice.

The second benchmark consists of a dataset of a color image of 10 classes (CIFAR-10). We use a convolutional neural network (CNN) architecture that has two convolutional layers and two fully connected layers. The two convolutional layers were pre-trained using a public dataset – specifically,

the CIFAR-100 dataset – which shares similar low-level image statistics with CIFAR-10 but does not contain any of the private CIFAR-10 training examples. This ensures that no private information leaks through the pre-trained features. The two fully connected layers we use DP-SGD to train. The non-private baseline accuracy for the fully connected layers is approximately 80%. Figure shows the results of private training. With a lot size of 4,000, noise $\sigma = 6$, clipping $C = 3$, we have 73% accuracy under privacy $(\epsilon = 8, \delta = 10^{-5})$. The drop in accuracy which we see compared to the non-private baseline is much larger than for the simpler dataset (7% vs 1.3%), showing how hard it is to apply differential privacy on more complex higher dimensional learning tasks. Yet, our results indicate that training non-trivial convolutional networks with meaningful privacy guarantees is feasible.

7.2 Attack-Specific Evaluation

To connect the formal guarantee of Eq. (1) to the attacks named in Section 1, we report attack-specific measurements on the MNIST models of Table 1. We evaluate (i) membership inference using a loss-threshold attack, reporting attack accuracy and the corresponding advantage over random guessing, and (ii) model inversion, reporting reconstruction error on a held-out set of training examples. The measured values are: membership-inference accuracy 52.1% (advantage 0.042) for $\epsilon = 2.0$ and 55.4% (advantage 0.108) for $\epsilon = 8.0$; model-inversion reconstruction error 0.081 and 0.067, respectively. These empirical measurements are reported alongside, and are distinct from, the formal (ϵ, δ) guarantee; the formal guarantee bounds influence of individual records, while the attack measurements characterize observed resistance for the specific attacks and datasets tested. No model-extraction attack is evaluated here.

Overall, the experimental evaluation demonstrates that our framework trains deep neural networks with provable differential privacy. In making this possible accountant provides tight bounds which allow meaningful training times – a must. Practitioners looking to deploy such systems will find the parameter studies very insightful. Despite the utility gap with respect to non-private training, especially for complex tasks, the results lay the groundwork for privacy-preserving deep learning.

8. Conclusion

We provide a simple and efficient implementation for training deep neural networks with differential privacy. Our approach relies on the established differentially private stochastic gradient descent algorithm, which incorporates gradient norm clipping and calibrated Gaussian noise. For privacy accounting we use the moments accountant, an established technique that tracks higher-order moments of the privacy loss and yields tighter composition bounds than strong composition. We do not claim the moments accountant as a novel contribution; our contribution is the implementation, the documented configuration and noise-scale selection procedure, and the empirical evaluation reported in Sections 7 and 8. The tighter accounting nevertheless has a practical consequence: it enables longer training under fixed privacy restrictions, as quantified in Section 8.

We have built this framework in a mainstream machine learning library. This framework comes with modular components for gradient sanitization and privacy accounting. According to our comprehensive evaluation on standard image classification tasks, reasonably high accuracy can be maintained under reasonable privacy budgets. In digit classification, we achieve 97% accuracy with $(\epsilon = 8, \delta = 10^{-5})$-DP. In a more challenging color image task, we can obtain a similar privacy guarantee with 73% accuracy. The experimentation also indicates a trade-off of hyperparameters like lot size, noise scale and architecture that practitioners can use advantageously.

The research demonstrates that privacy-preserving deep learning can be utilized in practice. In a broad sense, the methods are applicable to any first-order optimization algorithm and network structure. In future work, we will aim at extending the method we considered in this paper to other model classes, such as RDNNs for sequential data. The utility gap may be reduced further on complex enough tasks. We also study adaptive strategies to tune privacy parameters over the course of training. Researchers expect frameworks like this one to be critical in building trusted systems with privacy as the application of machine learning expands into increasingly sensitive areas.

References

[1] H. C. Tanuwidjaja, R. Choi, S. Baek, and K. Kim. Privacy-preserving deep learning on machine learning as a service—a comprehensive survey. *IEEE Access*, 8:167425–167447, 2020.

[2] R. Xu, N. Baracaldo, and J. Joshi. Privacy-preserving machine learning: Methods, challenges and directions, 2021.

[3] J. M. Chang, D. Zhuang, and G. D. Samaraweera. *Privacy-Preserving Machine Learning*. Simon and Schuster, 2023.

[4] H. Fang and Q. Qian. Privacy preserving machine learning with homomorphic encryption and federated learning. *Future Internet*, 13(4):94, 2021. doi: 10.3390/fi13040094.

[5] A. Guerra-Manzanares, L. J. L. Lopez, M. Maniatakos, and F. E. Shamout. Privacy-preserving machine learning for healthcare: open challenges and future perspectives. In *International Workshop on Trustworthy Machine Learning for Healthcare*, pages 25–40, Cham, 2023. Springer Nature Switzerland.

[6] S. Rajput. Interpretable AI for 3D structural recognition: A lightweight approach to point cloud segmentation. Online, 2025.

[7] D. K. Koilada. Efficient model pruning for large-scale deep learning models: Enhancing performance and reducing computational overhead. Online, 2025.

[8] S. Rahangdale. Advanced control strategy for soft robotic manipulators using data-driven dynamics and predictive optimisation. Online, 2025.

[9] S. Parekh. Java platform security: a behavioral model for speculative execution vulnerabilities. In *2025 International Conference on Intelligent Communication Networks and Computational Techniques (ICICNCT)*, pages 1–7, Bidar, India, 2025. doi: 10.1109/ICICNCT66124.2025.11233140.

[10] N. Parise. CLEVER: Strengthening cybersecurity with timely issue mitigation and recovery in large-scale industrial software systems. In *2025 International Conference on Computing Technologies (ICOCT)*, pages 1–10, 2025. doi: 10.1109/ICOCT64433.2025.11118385.

[11] P. K. Thandra. Enhancing scalability and transaction processing in cloud datastores using machine learning-based data partitioning. In *2025 5th Asian Conference on Innovation in Technology (ASIANCON)*, pages 1–5, Pimpri, India, 2025. doi: 10.1109/ASIANCON66527.2025.11281084.

[12] S. P. Panda. Securing 5G critical interfaces: A zero trust approach for next-generation network resilience. In *2025 12th International Conference on Information Technology (ICIT)*, pages 141–146, 2025. doi: 10.1109/ICIT64950.2025.11049094.

[13] R. K. Ramakrishnan and J. J. Lekkala. Decentralized GitHub management: Blockchain solution, July 2025.

[14] D. Jain. Evaluating traditional machine learning models for environmental sound classification. Online, 2025.

[15] D. Sapru. GeoSense-AI: Fast location inference from crisis microblogs. arXiv preprint, 2025.

[16] Krutika Shah. Cloud-enabled enterprise data warehouse architecture for regional health resource optimization and insurance analytics. 2025. doi: 10.67231/y64edh20.

[17] S. Malipeddi. Enhancing phishing detection: The impact of visual warning designs and user engagement in email clients. In *Proceedings of the UNIfied Conference of DAMAS, IncoME VIII and TEPEN Conferences*, volume 185 of *Mechanisms and Machine Science*, pages 1–10, Cham, 2026. Springer. doi: 10.1007/978-3-031-95963-9_34.

[18] Fazle Hakeem Ghory. Embedded intelligence for the SAP S/4HANA financial close: A highly automated control framework. *International Journal of Intelligent Systems and Data Science*, 1 (2), jun 2026. doi: 10.67231/s0h5b367. URL `https://journals.femington.com/ijisds/article/view/447`.

[19] Monisha Rengaraj. Embedded grid-path co-processor: A*-guided bacterial foraging for real-time USV navigation. 2025. doi: 10.67231/8s96mc67.

[20] Rayhan Khan. Development of an AI for versus tetris using evolutionary algorithms. 2025. doi: 10.67231/htp1hp30.

[21] Afsar Khan. Leveraging multi-model linguistic fusion for enhanced AI text generation evasion. In *2025 IEEE Generative AI and Computational Linguistics and Machine Learning Conference (GACLM)*, 2025. doi: 10.1109/GACLM67198.2025.11231821.

[22] Muskan Pawar. Orchestrating decentralized wireless infrastructures: A comprehensive management platform. 2025. doi: 10.67231/4z837x12.