Scientific Reports

https://doi.org/10.1038/s41598-026-44009-2

Article in Press

Stabilizing updates in differentially private stochastic gradient descent with buffered rejection

Received: 14 November 2025

Accepted: 9 March 2026

Published online: 18 March 2026

Cite this article as: Deng S., Zhang K., Zhang W. *et al.* Stabilizing updates in differentially private stochastic gradient descent with buffered rejection. *Sci Rep* (2026). https://doi.org/10.1038/s41598-026-44009-2

Sifan Deng, Kai Zhang, Weilin Zhang, Huiqin Jiang & Pei-Wei Tsai

Stabilizing Updates in Differentially Private Stochastic Gradient Descent with Buffered Rejection

Sifan Deng[1], Kai Zhang[1,*], Weilin Zhang[2], Huiqin Jiang[1], and Pei-Wei Tsai[3]

[1]School of Computer and Information Engineering, Xiamen University of Technology, Xiamen, 361024, China
[2]New Energy Engineering Design & Research Institute, China United Engineering Corporation Limited, Hangzhou, 310052, China
[3]Department of Computing Technologies, Swinburne University of Technology, Melbourne, 3122, Australia
[*]Corresponding Author: Kai Zhang, Email: kaizhang@xmut.edu.cn

ABSTRACT

Differentially private stochastic gradient descent is a standard algorithm for training deep models on sensitive data, but under tight privacy budgets it must add large noise to every step, which slows convergence and reduces accuracy. Selective update methods for differential private stochastic gradient descent reject updates that fail a noisy validation test and save privacy cost, but each decision still relies on a single noisy signal and remains unstable. We propose a differential private training algorithm that combines a buffered rejection mechanism with a phased parameter decay strategy for stochastic gradient descent. In each iteration, the proposed algorithm maintains two candidate updates, evaluates their privately perturbed loss improvements, and applies a local preferential choice. This buffered comparison spends privacy budget on directions that are more likely to be beneficial. The phased decay strategy tracks validation accuracy and gradually adjusts the noise multipliers, learning rate, and rejection threshold to match the current training stage. Experiments on MNIST, Fashion-MNIST, CIFAR-10, and IMDb with identical privacy budgets show that the proposed algorithm consistently improves test accuracy over the standard differential private stochastic gradient descent and the selective update based differential private stochastic gradient descent, typically by 0.5–2 percentage points, and converges faster at the same privacy level. Membership inference evaluations report area under the ROC curve values close to 0.5, indicating that these gains do not weaken empirical privacy.

Keywords: Differential Privacy, Deep Learning, Buffered Rejection Mechanism

Introduction

Deep learning is now widely used in computer vision[1–3], natural language processing[4,5], and pattern recognition[6–8]. As these systems are increasingly trained on sensitive data, there is a growing need to protect individual privacy without sacrificing model accuracy. Differential privacy (DP)[9] provides a principled way to control information leakage from training data. In practice, most deep learning frameworks implement DP through stochastic gradient based training, where each update is perturbed by carefully calibrated noise[10].

The standard DP-SGD algorithm[11] often suffers from high-variance gradient estimates and degraded model utility under strong privacy constraints. It clips per-sample gradients to bound sensitivity and then adds Gaussian noise to the aggregated gradient at every iteration. When the privacy budget ϵ is small, the required noise scale becomes large. The model then suffers from slow convergence and a noticeable loss in utility compared to non-private training. To reduce unnecessary privacy consumption, Differentially Private Stochastic Gradient Descent with Selective Update and Release[12] (DPSUR), a typical selective update method for differential private stochastic gradient descent, introduces a rejection mechanism.After computing a noisy gradient update, it evaluates the loss on a validation batch. If the noisy loss does not decrease, the update is rejected and the model parameters remain unchanged. In this way, the method avoids spending privacy budget on iterations that appear unhelpful. However, this decision is based on a single noisy observation of the loss difference. In high-noise regimes, this single signal is unstable. The algorithm can accept suboptimal updates or reject truly beneficial ones, which leads to inefficient use of the privacy budget.

To address this problem, we propose Differentially Private Stochastic Gradient Descent with Buffering and Rejection (DPSGD-BR). The key idea is to replace the single noisy decision with a local preferential selection between multiple candidate updates. In each iteration, DPSGD-BR maintains a small buffer of candidate updates that have passed a private rejection test. The algorithm then compares their noisy loss improvements and selects the candidate that is more likely to reduce the true loss. This buffered selection improves robustness to gradient noise while keeping the rejection-based saving of privacy budget. DPSGD-BR also includes an adaptive mode switching mechanism. When the algorithm observes a sequence of rejected candidates, it temporarily switches from a dual-buffer mode to a single-update mode to avoid long stagnation near flat or local

regions. In addition, a phased parameter decay strategy adjusts the noise multipliers, learning rate, and rejection threshold over time. These adjustments seek a better balance between exploration under high noise and stable convergence under a fixed privacy budget. Figure 1 illustrates the qualitative effect of these changes. The optimization path of DPSGD-BR follows a smoother trajectory and reaches lower-loss regions than the path of vanilla DP-SGD under the same noise level.

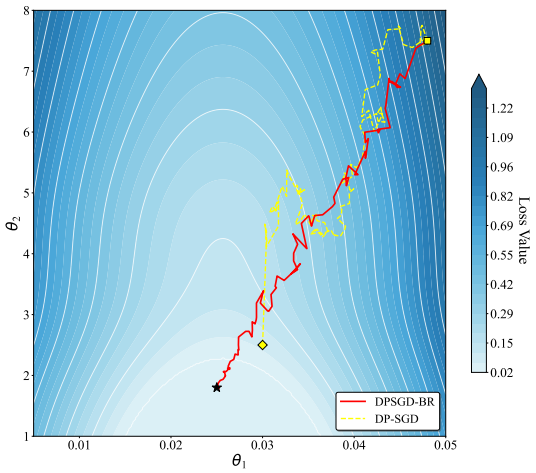

Figure 1. Optimization Paths on the Loss Landscape

We evaluate DPSGD-BR on several standard image and text benchmarks under matched privacy budgets. The experiments show that DPSGD-BR consistently improves test accuracy over DP-SGD and DPSUR, with faster convergence and more stable training curves. On average, the gains are about 0.5% in test accuracy under the same ϵ. These results suggest that more reliable update selection can significantly improve privacy budget utilization in DP training.

The main contributions of this paper are summarized as follows:

- We propose DPSGD-BR, a buffered rejection framework that replaces the single noisy decision of DPSUR[12] with a local preferential selection between candidate updates, which reduces the impact of noise on the accept–reject decision.
- We introduce an adaptive mode switching mechanism and a phased parameter decay strategy that jointly control the noise multipliers, learning rate, and rejection threshold to improve convergence under a fixed privacy budget.
- We provide a privacy analysis of DPSGD-BR under the Rényi DP framework and show that the method preserves the overall DP guarantee while reducing the number of privacy consuming updates.
- The proposed algorithm achieves consistent improvements in test accuracy and privacy budget utilization compared with the standard DP-SGD[11] and DPSUR[12] under the same privacy budget constraints on multiple public datasets covering computer vision and natural language processing tasks.

Related Work

Research on the use of differential privacy (DP)[9] in deep learning[13] mainly builds on the DP-SGD framework[11]. DP-SGD clips per-sample gradients and adds Gaussian noise to the aggregated update at every iteration. This procedure provides a clear (ϵ, δ)-DP guarantee, but it also increases the variance of gradient estimates in high-noise settings. As a result, convergence becomes slower and model utility is often inferior to non-private training. Existing work has addressed these issues along two main lines.

Optimizing Gradient Estimation and Update Strategies

One line of work aims to improve the quality of gradient updates and the efficiency of training under DP noise. Acceleration and noise control mechanisms adjust how updates are computed so that the model can converge with fewer steps. For example, methods in[14,15] modify sampling and training dynamics to reduce the impact of noisy gradients. Other studies focus on adaptive optimization under DP[16–21]. They tune learning rates, clipping thresholds, or noise multipliers based on the training process in order to reduce variance and stabilize updates. These approaches share a key property. They all follow the principle of consuming privacy budget at every iteration[22]. The methods improve the quality of each update, but they do not reduce the number of updates that spend privacy on noisy steps that bring little or no benefit to the model.

Our method follows the same DP-SGD backbone but changes how updates are selected and scheduled. By combining buffered candidate updates with a phased decay of key parameters, it aims to reduce the number of privacy-consuming steps while keeping or improving the utility of accepted updates.

Selective Update Mechanisms Based on Validation and Rejection

A second line of work seeks to control the update frequency and thus reduce privacy budget consumption. The goal is to skip updates that are likely to be dominated by noise and to keep updates that have a clear positive effect on model performance. The most representative method in this direction is DPSUR (Differentially Private Stochastic Gradient Descent with Selective Update and Release)[12]. In each iteration t, DPSUR first computes a private estimate of the loss change on a validation batch. It measures the difference between the loss before and after the candidate update and then adds Gaussian noise to this difference. The update is accepted only when the noisy loss change falls below a threshold Z. In that case, the method applies the update and accounts for the privacy cost. Otherwise, the update is rejected and the model parameters remain unchanged. This rejection mechanism has been shown to reduce unnecessary privacy expenditure and to speed up convergence by filtering out updates that appear unhelpful[12].

The main limitation of DPSUR is that the accept–reject decision is based on a single noisy observation of the loss difference. In high-noise regimes, this single signal is unstable. The algorithm can accept suboptimal or even harmful updates and can also reject updates that would have improved the true loss. This instability restricts the further improvement of model utility and privacy budget efficiency. It is also related to known privacy risks, since inaccurate decisions on noisy outputs can leave room for attacks that exploit model behaviour under DP training[23,24]. These observations motivate the need for more reliable selection mechanisms that still operate within the DP accounting framework.

DPSGD-BR is designed in response to this limitation. It replaces the single noisy decision with a buffered comparison of multiple candidates and couples this with an adaptive mode switching and decay strategy. In the next section we describe how this design stabilizes the selection of updates and improves the use of the privacy budget under the same DP constraints.

Preliminaries

Differential privacy (DP)[9,25,26] provides the mathematical basis for most private training algorithms, including DP-SGD and its variants. In this section, we briefly review the standard definition of DP, the Gaussian mechanism and sensitivity, and the Rényi differential privacy (RDP) framework[27,28]. These tools are used later to analyze the privacy guarantee and budget consumption of the proposed DPSGD-BR method.

Differential Privacy

DP formalizes the idea that the output of a randomized algorithm should not change much when a single record in the dataset is modified[9]. It quantifies the worst-case privacy loss for any pair of adjacent datasets. We first present the standard (ϵ, δ)-DP definition as follows.

Definition 1 (**(ϵ, δ)-Differential Privacy**) [29–31] *A randomized algorithm $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ satisfies (ϵ, δ)-differential privacy if for any pair of adjacent datasets $D, D' \in \mathcal{D}$ that differ in one record and for any measurable set $S \subseteq Range(\mathcal{M})$, the following inequality holds*

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

where ϵ is the main privacy parameter and measures the strength of the guarantee. A smaller ϵ means stronger privacy. The parameter δ is a small failure probability that allows rare events where the guarantee can be violated. In the design of private optimization algorithms, we aim to keep ϵ as small as possible by tracking and limiting privacy budget consumption over all training steps.

Gaussian Mechanism and Sensitivity Analysis

Many private training procedures involve numerical queries such as sums of gradients. The Gaussian mechanism[31,32] is a standard way to release such queries under DP. Its noise scale depends on the sensitivity of the query. We present the notion of L_2 sensitivity as follows.

Definition 2 (**L_2 Sensitivity**) *For a function $f : \mathcal{D} \to \mathbb{R}^d$, the L_2 sensitivity $\Delta_2 f$ is the maximum change in its output over all adjacent datasets D and D':*

$$\Delta_2 f = \max_{D, D'} \|f(D) - f(D')\|_2. \tag{2}$$

In the DP-SGD algorithm, per-sample gradient clipping with threshold C enforces a bound on the L_2 sensitivity of the summed gradient. This step ensures that no single example can have an unbounded influence on the update. The Gaussian mechanism releases $f(D)+\mathcal{N}(0,\sigma^2 I)$ and can be calibrated to satisfy (ϵ,δ)-DP. A common sufficient condition is

$$\sigma \geq \frac{\sqrt{2\ln(1.25/\delta)}}{\epsilon}. \tag{3}$$

This relationship exposes a basic tension in private optimization. Stronger privacy, that is a smaller target ϵ, requires a larger noise scale σ. In iterative training, every gradient update is perturbed by high noise, which increases gradient variance and can degrade both convergence speed and model utility.

RDP Composition Analysis and Privacy Budget Tracking

Training algorithms such as DP-SGD and DPSGD-BR add noise during each iteration on the same dataset. The overall privacy loss is cumulative and must be tracked over all iterations. To obtain a tighter and more convenient analysis than classical DP composition, we use the RDP framework[27,28].

Definition 3 (Rényi Differential Privacy (RDP)) *A randomized algorithm $\mathcal{M}$ satisfies (α,ρ)-RDP for order $\alpha \in (1,\infty)$ if for any two adjacent datasets D and D' the Rényi divergence of order α between the output distributions is bounded*

$$D_\alpha(\mathcal{M}(D)\,\|\,\mathcal{M}(D')) \leq \rho, \tag{4}$$

where ρ is the RDP budget at order α. The Rényi divergence (the theoretical foundation of RDP) is defined as: for two probability distributions P and Q over the sample space $\mathcal{X}$ (with P absolutely continuous with respect to Q, i.e., $Q(x)=0$ implies $P(x)=0$), the Rényi divergence of order $\alpha \in (1,\infty)$ is

$$D_\alpha(P\|Q) = \frac{1}{\alpha-1}\ln\left(\mathbb{E}_{x\sim Q}\left[\left(\frac{P(x)}{Q(x)}\right)^\alpha\right]\right), \tag{5}$$

where $\mathbb{E}$ denotes the expectation operator, ln *is the natural logarithm.*

A key advantage of RDP is its simple composition rule. Suppose a sequence of mechanisms $\mathcal{M}_1,\ldots,\mathcal{M}_k$ is applied to the same dataset. If each $\mathcal{M}_i$ satisfies (α,ρ_i)-RDP, then the composed mechanism satisfies (α,ρ_{total})-RDP with

$$\rho_{total} = \sum_{i=1}^{k}\rho_i. \tag{6}$$

This additive property makes it convenient to track privacy loss in iterative training.

In our setting, each accepted update in DPSGD-BR contributes an RDP cost from the noisy gradient and from the noisy loss-based selection. By rejecting some candidate updates, the algorithm reduces the total number of privacy consuming steps and thus lowers the cumulative RDP budget ρ_{total}. In later sections we use this framework to convert the RDP accounting back into an overall (ϵ,δ) guarantee for the full training process.

The Proposed DPSGD-BR Method

This section presents the proposed DPSGD-BR algorithm. Different with the DPSUR[12], DPSGD-BR introduces a buffered rejection mechanism that compares two candidate updates before accepting one, and a phased parameter decay strategy that adapts the noise and learning rate to the current training stage.

Problem Setting and Baseline

We consider standard supervised training with a model parameter vector W, a private dataset $\mathcal{D}$, and a loss function $\mathcal{L}$. Training proceeds in iterations. At each iteration a minibatch is sampled, gradients are computed, and a noisy update is applied under DP constraints. DP-SGD[11] clips per-sample gradients to a bound C, aggregates them, and adds Gaussian noise with variance controlled by a multiplier σ_t. Every update consumes part of the privacy budget. DPSUR[12] builds on this pipeline. After computing a noisy update, it evaluates the loss on a validation batch and uses a noisy estimate of the loss difference to decide whether to accept the update. If the noisy loss does not improve beyond a threshold, the update is rejected and the parameters remain unchanged. This mechanism reduces privacy spending on clearly unhelpful updates. However, the accept or reject decision relies on a single noisy loss difference, which can be unstable when the noise scale is large. DPSGD-BR follows the

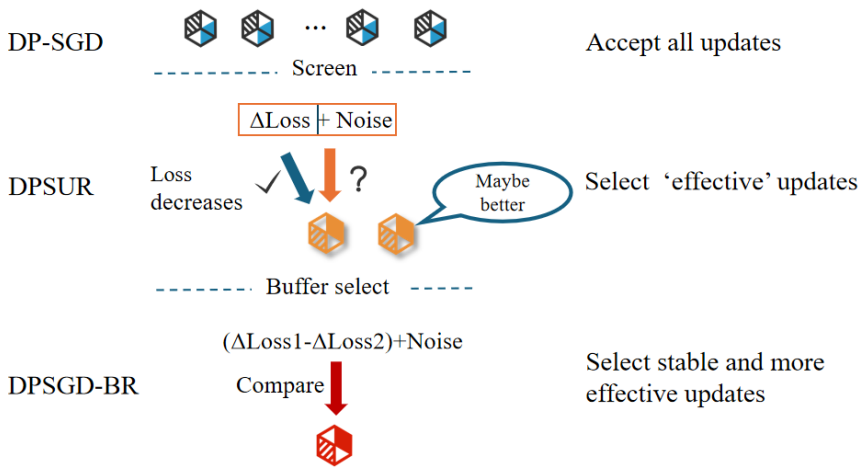

Figure 2. Core Differences in Update Decision Logic: DP-SGD vs. DPSUR vs. DPSGD-BR

same basic training setting as DP-SGD and DPSUR. Its goal is to make better use of the privacy budget by stabilizing the update decision and by adapting training parameters to the progress of the model.

In addition, to clarify the core differences between DPSGD-BR and DPSUR, a schematic diagram (Figure 2) is provided to visually contrast the update decision logic of three algorithms (DP-SGD, DPSUR, and DPSGD-BR), where all designs of DPSGD-BR are targeted at addressing the inherent limitations of DPSUR.

The schematic diagram clearly illustrates:

- DP-SGD[11] performs parameter updates unconditionally in each iteration without any selective judgment.
- DPSUR [12] judges whether to perform parameter updates based on the loss change after a single noise perturbation, which is prone to misjudgment due to noise interference.
- DPSGD-BR reduces noise interference and improves privacy budget utilization efficiency by comparing the noise-perturbed loss differences of two candidate updates through a dual-candidate buffer mechanism and selecting the optimal update direction.

The Overall workflow of DPSGD-BR

Figure 3 shows the overall workflow of DPSGD-BR. Each iteration can be divided into four stages:

- generate candidate updates with different noise realizations;
- filter each candidate using a private estimate of the loss change on a validation batch;
- select one update from the buffer or switch to a degraded mode when many candidates are rejected;
- apply the accepted update and adjust key training parameters in a phased manner.

In the standard mode, the algorithm maintains a buffer that holds two valid candidate updates. Both candidates must first pass a private rejection filter. Once the buffer is full, the algorithm compares their noisy loss improvements and selects one for application. The privacy budget is charged only when an update is applied. When the training process encounters a stretch of consecutive rejections, the algorithm activates a degradation mode. In this mode only one candidate is required and any candidate that passes the filter can be applied directly. This avoids long periods without parameter changes.

The phased parameter decay strategy runs on top of this buffered update procedure. It monitors the change in validation accuracy between accepted updates. When the model improves quickly, the strategy decays the noise multipliers and learning rate more aggressively. When the improvement slows down, it uses slower decay and gradually relaxes the rejection threshold. In later sections we show that this combination improves both convergence and privacy budget usage.

Buffered Rejection Mechanism

The buffered rejection mechanism is the main difference between DPSGD-BR and DPSUR[12]. It still decides whether to accept or reject updates using a noisy validation loss, but it relies on two candidates and a preferential selection rule instead of a single noisy decision.

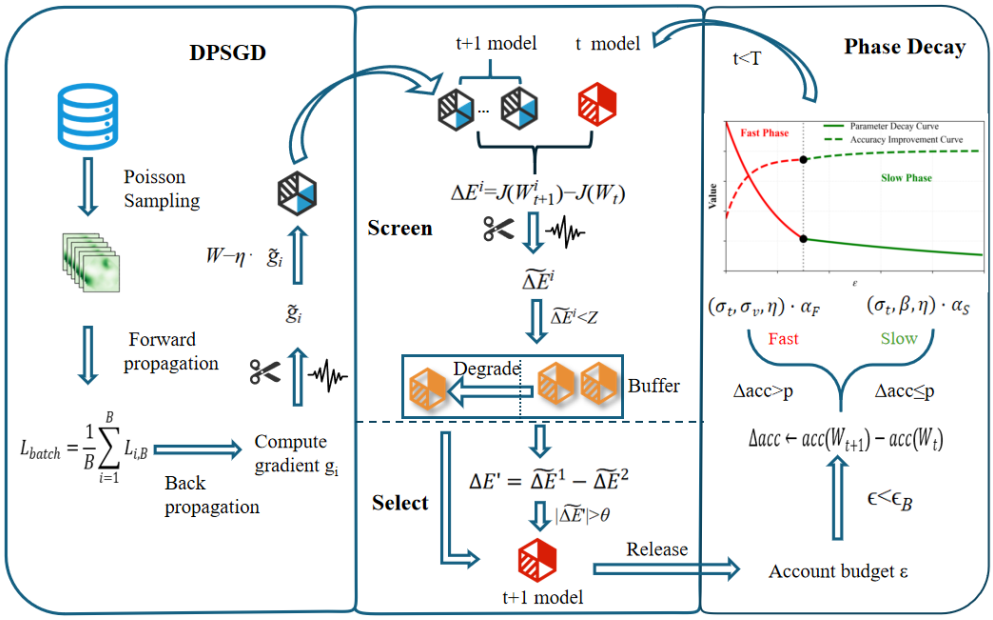

Figure 3. Workflow of DPSGD-BR

Candidate Generation and Private Loss Filtering

We first recall the DP-SGD style gradient step that DPSGD-BR uses to construct each candidate update.

Definition 4 (Candidate update in DP-SGD[11]) *Given W_t and a training batch $\mathcal{B}$ sampled from $\mathcal{D}$, the gradient of the loss is*

$$g = \nabla_W \mathcal{L}(W_t, \mathcal{B}). \tag{7}$$

The gradient is clipped to norm C and perturbed by Gaussian noise with multiplier σ_t:

$$\tilde{g} = Clip(g, C) + \mathcal{N}(0, \sigma_t^2 C^2 I). \tag{8}$$

The clipped and noisy gradient yields a candidate update

$$W_{t+1}^i = W_t - \eta \tilde{g}, \tag{9}$$

where η is the learning rate and $i \in \{1, 2\}$ indexes the candidate in the buffer.

Based on this candidate, DPSGD-BR uses a private loss based filter to decide whether the update should enter the buffer.

Definition 5 (Private noisy loss change) *Let $\mathcal{B}_v$ be a validation batch sampled from $\mathcal{D}$. The losses before and after applying the candidate are*

$$J(W_t) = \mathcal{L}(W_t, \mathcal{B}_v), \quad J(W_{t+1}^i) = \mathcal{L}(W_{t+1}^i, \mathcal{B}_v). \tag{10}$$

The raw loss change is

$$\Delta E^i = J(W_{t+1}^i) - J(W_t). \tag{11}$$

This value is clipped to a bound C_v and perturbed by Gaussian noise with variance controlled by σ_v:

$$\widetilde{\Delta E}^i = Clip(\Delta E^i, C_v) + \mathcal{N}_{loss,i}, \tag{12}$$

where $\mathcal{N}_{loss,i} \sim \mathcal{N}(0, \sigma_v^2)$.

The algorithm then uses this private loss change to decide whether the candidate is stored for later comparison.

Definition 6 (Buffered acceptance rule) *Let the threshold Z be defined as*

$$Z = \beta C_v, \tag{13}$$

where $\beta < 0$ controls the strictness of the filter. A candidate $(W_{t+1}^i, \widetilde{\Delta E}^i)$ is accepted into the buffer if

$$\widetilde{\Delta E}^i < Z. \tag{14}$$

If the condition holds, the pair $(W_{t+1}^i, \widetilde{\Delta E}^i)$ is stored in the buffer and the consecutive rejection counter is reset to zero. Otherwise the candidate is discarded and the counter is increased by one. While the buffer is being filled, privacy budget is not charged, since these noisy loss values are internal to the selection mechanism.

Preferential Selection and Degradation

Once the buffer holds two valid candidates, DPSGD-BR compares their noisy loss improvements and selects one update to apply. We formalize this preferential selection rule.

Definition 7 (Preferential selection rule) *Let $(W^1, \widetilde{\Delta E}^1)$ and $(W^2, \widetilde{\Delta E}^2)$ be the two candidates stored in the buffer. Their difference in noisy loss improvement is*

$$\Delta E' = \widetilde{\Delta E}^1 - \widetilde{\Delta E}^2. \tag{15}$$

Given a difference threshold $\theta > 0$, the next model parameters W_{t+1} are chosen as

$$W_{t+1} = \begin{cases} W^1, & \text{if } \Delta E' > \theta, \\ W^2, & \text{if } \Delta E' < -\theta, \\ Random(W^1, W^2), & \text{if } |\Delta E'| \le \theta. \end{cases} \tag{16}$$

Only one update is applied in each iteration, and the privacy accountant is advanced once for this accepted update.

The buffered selection reduces sensitivity to a single noisy observation, since both candidates are evaluated with independent noise and their comparison cancels part of the randomness. In high noise regimes the filter can still reject many candidates in a row, which motivates a degraded mode for robustness.

Definition 8 (Degradation condition and single-candidate mode) *Let $count_{reject}$ denote the number of consecutive rejected candidates, and let T_{max} be a preset integer threshold. The algorithm enters a degraded mode when*

$$count_{reject} \ge T_{max}. \tag{17}$$

In this mode the required buffer size is set to $n = 1$. Any candidate that satisfies the buffered acceptance rule $\widetilde{\Delta E}^i < Z$ is applied immediately as W_{t+1} without waiting for a second candidate. The counter $count_{reject}$ is reset to zero whenever a candidate is accepted, and the algorithm can later return to the two-candidate buffered mode.

The degradation mechanism keeps the training process moving under strict filtering and heavy noise. It also avoids very long stretches without updates, which reduces the risk that an observer could infer information from the pattern of rejections.

Phased Parameter Decay Strategy

The buffered rejection mechanism improves how updates are chosen. DPSGD-BR also adapts several training parameters over time so that the optimization better matches the current learning stage. The phased parameter decay strategy controls the learning rate η, the gradient noise multiplier σ_t, the validation noise multiplier σ_v, and the rejection threshold β. We describe how the strategy measures progress between accepted updates as follows.

Definition 9 (Validation accuracy change) *Let $acc(W)$ denote the accuracy on a held out validation set. After an accepted update from W to W_{new}, the change in validation accuracy is*

$$\Delta acc = acc(W_{new}) - acc(W). \tag{18}$$

A positive threshold p divides the training process into two regimes: $\Delta acc > p$ indicates a fast improvement stage, and $\Delta acc \le p$ indicates a slower regime near a convergence plateau.

The decay rules in DPSGD-BR are then defined in terms of the two phases determined by Δacc as follows.

Definition 10 (Fast decay phase) *When $\Delta acc > p$, DPSGD-BR is in the fast decay phase. The gradient noise multiplier, validation noise multiplier, and learning rate are updated with a decay factor $\alpha_F \in (0,1)$:*

$$(\sigma_t, \sigma_v, \eta) \leftarrow (\sigma_t, \sigma_v, \eta) \cdot \alpha_F. \tag{19}$$

The threshold β remains unchanged in this phase.

This rule reflects the observation that early in training both gradients and noise are large and the quality of different updates varies widely. A faster decay of σ_t, σ_v, and η helps the algorithm move out of the very high noise regime and reach a region where the signal to noise ratio is more favorable, while a fixed β keeps the filter relatively strict.

Definition 11 (Slow decay phase) *When $\Delta acc \leq p$, DPSGD-BR is in the slow decay phase. The gradient noise multiplier, rejection threshold, and learning rate are updated with a slower decay factor $\alpha_S \in (0,1)$:*

$$(\sigma_t, \beta, \eta) \leftarrow (\sigma_t, \beta, \eta) \cdot \alpha_S. \tag{20}$$

The validation noise multiplier σ_v stays fixed in this phase.

The slower decay prevents the parameters from shrinking too quickly when the model is already close to convergence. The gradual decrease of β relaxes the rejection condition and keeps a sufficient number of accepted updates in difficult regions of the loss surface. Keeping σ_v fixed avoids an extremely small noise level on the validation loss, which would otherwise lead to faster privacy budget consumption in the validation mechanism under RDP composition. In practice, the decay strategy is also constrained by an upper bound on the total privacy budget.

Definition 12 (ϵ-boundary stopping rule) *Let ϵ denote the accumulated privacy budget during training and let ϵ_B be a preset boundary. When*

$$\epsilon \geq \epsilon_B, \tag{21}$$

all parameter decay is stopped and the remaining iterations are run with fixed $(\sigma_t, \sigma_v, \beta, \eta)$. The value of ϵ_B is chosen to balance experimental efficiency and coverage of higher privacy budgets.

This boundary allows DPSGD-BR to explore a range of ϵ values in experiments without requiring an excessive number of iterations at very small noise levels.

Privacy Analysis under RDP

We now analyze the privacy guarantee of DPSGD-BR using the RDP framework introduced in Preliminaries Section. The main idea is to treat each accepted update as the composition of two subsampled Gaussian mechanisms, one for the noisy gradient step and one for the noisy validation loss based filter. We first recall the RDP cost of a subsampled Gaussian mechanism.

Definition 13 (RDP cost of subsampled Gaussian mechanism[27,28]) *Let q be the subsampling rate and let σ be the noise multiplier. The RDP cost at order $\alpha > 1$ is*

$$R(\alpha, q, \sigma) = \frac{1}{\alpha - 1} \ln \left(\sum_{i=0}^{\alpha} \binom{\alpha}{i} (1-q)^{\alpha - i} q^i \exp\left(\frac{i^2 - i}{2\sigma^2} \right) \right). \tag{22}$$

where i is a summation variable, representing the number of selected samples in sampling. It is used to iterate through all sampling scenarios and perform a weighted calculation of the privacy loss for a single model update.

Definition 14 (RDP cost of one DPSGD-BR update) *Let q_{train} and σ_t be the subsampling rate and noise multiplier for the gradient update, and let q_{val} and σ_v be those for the validation loss. The RDP cost of one accepted update at order α is*

$$R_{update}(\alpha) = R(\alpha, q_{train}, \sigma_t) + R(\alpha, q_{val}, \sigma_v). \tag{23}$$

The total privacy loss over the whole training run depends on how many such updates are accepted.

Definition 15 (Total RDP budget of DPSGD-BR) *Let N_{accept} be the number of accepted updates during training. All rejected candidates are filtered internally and do not incur extra privacy cost. By the additive property of RDP composition[27,28,33], the total RDP budget at order α is*

$$R_{total}(\alpha) = N_{accept} \cdot R_{update}(\alpha). \tag{24}$$

This expression shows that the per update privacy cost of DPSGD-BR matches that of DPSUR, while the buffered rejection mechanism improves privacy budget utilization through a more informative pattern of accepted updates.

To express the guarantee in the standard (ϵ, δ) form, we use a known conversion from RDP to (ϵ, δ)-DP.

Definition 16 (Conversion from RDP to (ϵ, δ)-DP[34]) *Let $R_{total}(\alpha)$ be the total RDP budget at order $\alpha > 1$ and let $\delta \in (0, 1)$ be a target failure probability. The corresponding privacy parameter $\epsilon(\alpha)$ satisfies*

$$\epsilon(\alpha) = R_{total}(\alpha) + \ln\left(\frac{\alpha - 1}{\alpha}\right) - \frac{\ln\delta + \ln\alpha}{\alpha - 1}. \tag{25}$$

The tightest bound is obtained by searching over a range of orders α and taking the smallest $\epsilon(\alpha)$, which is implemented by a standard RDP accountant.

Since N_{accept} is usually much smaller than the total number of iterations T, DPSGD-BR can reach a given privacy budget ϵ with more informative updates than DP-SGD and DPSUR, while maintaining the same formal privacy accounting framework.

Algorithm Summary and Takeaway

Algorithm 1 summarizes the full DPSGD-BR procedure. The algorithm loops over training iterations. In each iteration it uses the buffered rejection mechanism to generate and filter candidate updates, applies one accepted update to the model, accounts for privacy loss, and then adjusts parameters according to the phased decay strategy. The main takeaway is that DPSGD-BR spends privacy budget only on updates that clear a double filter and prefers the candidate with stronger evidence of loss reduction. Combined with stage-aware parameter decay, this design allows the model to gather more informative updates under the same privacy budget compared to standard DP-SGD style training.

Experimental Results and Discussion

This section evaluates DPSGD-BR on image and text classification tasks. We first describe the experimental setup and metrics, then present results on overall performance, convergence behavior, ablation studies, robustness to membership inference attacks, and hyperparameter sensitivity. A final discussion connects these findings back to the design choices in buffered rejection and phased parameter decay.

Experimental Setup

All experiments are implemented in PyTorch[10] with DP support provided by Opacus. Training is performed on a workstation equipped with NVIDIA RTX4090 and 64 GB system memory. The same code base and implementation details are used across all compared methods to ensure a fair comparison.

We evaluate DPSGD-BR on four standard benchmarks that cover both vision and language tasks:

- **MNIST**[35]: Handwritten digit recognition with 28×28 grayscale images and 10 classes.
- **Fashion-MNIST (FMNIST)**[36]: Clothing classification with the same input format and number of classes as MNIST but more complex visual patterns.
- **CIFAR-10**[37]: Natural image classification with 32×32 RGB images and 10 classes.
- **IMDb**[38]: Binary sentiment classification on movie reviews.

For each dataset we follow common practice in the DP-SGD literature[11,12,21,39]. Convolutional networks are used for MNIST, FMNIST, and CIFAR-10, and a recurrent or embedding based architecture is used for IMDb. All DP methods share the same privacy parameters. The failure probability is fixed at $\delta = 10^{-5}$ and the target privacy budget ϵ ranges from 1 to 4. For baselines we include vanilla DP-SGD[11], DPSGD-HF[39], DPSGD-TS[21], and DPSUR[12]. Each method uses its recommended hyperparameters from the original papers, with minor tuning when needed to adapt to our training schedule.

Evaluation Metrics

We focus on two aspects of performance: predictive utility and robustness to membership inference attacks (MIA). The main utility metric is test classification accuracy. The privacy side is evaluated through attack success in black-box and white-box MIA settings. To make notation explicit, we recall the metric used for classification accuracy. We formalize the definition of accuracy used in this work.

Algorithm 1: DPSGD-BR

Input: Training data $\mathcal{D}$, initial weights W_0, iterations T, batch sizes (B, B_v), dataset size N, loss $\mathcal{L}$, parameters $(\eta, C, \sigma_t, \sigma_v, \beta, C_v)$, thresholds $(Z, \theta, T_{\max}, p)$, decay factors (α_F, α_S), privacy boundary ϵ_B.
Output: Trained weights W_T.

Initialize $W \leftarrow W_0$, $\epsilon \leftarrow 0$, $\text{count}_{\text{reject}} \leftarrow 0$.
for $t = 1$ to T **do**
  Buffer $\leftarrow \emptyset$.
  $n \leftarrow (\text{count}_{\text{reject}} \geq T_{\max})?1 : 2$.
  while $|\text{Buffer}| < n$ **do**
    $\mathcal{B} \leftarrow \text{SampleBatch}(\mathcal{D}, B, N)$.
    $g \leftarrow \nabla_W \mathcal{L}(W, \mathcal{B})$.
    $\widetilde{g} \leftarrow \text{ClipAndNoiseGrads}(g, C, \sigma_t)$.
    $W_{\text{t}} \leftarrow W - \eta \cdot \widetilde{g}$.
    $\mathcal{B}_v \leftarrow \text{SampleValidationBatch}(\mathcal{D}, B_v, N)$.
    $J(W_{\text{t}}) \leftarrow \mathcal{L}(W_{\text{t}}, \mathcal{B}_v)$, $J(W) \leftarrow \mathcal{L}(W, \mathcal{B}_v)$.
    $\Delta E \leftarrow J(W_{\text{t}}) - J(W)$.
    $\widetilde{\Delta E}^{\text{i}} \leftarrow \text{ClipAndNoiseLosses}(\Delta E, C_v, \sigma_v)$.
    if $\widetilde{\Delta E}^{\text{i}} < Z$ **then**
      Buffer $\leftarrow$ Buffer $\cup \{(W_{\text{t}}, \widetilde{\Delta E}^{\text{i}})\}$, $\text{count}_{\text{reject}} \leftarrow 0$.
    else
      $\text{count}_{\text{reject}} \leftarrow \text{count}_{\text{reject}} + 1$.
  end while
  if $n = 1$ **then**
    $W_{\text{new}} \leftarrow \text{Buffer}[1]$.
  else
    $(W^1, \widetilde{\Delta E}^1), (W^2, \widetilde{\Delta E}^2) \leftarrow$ Buffer.
    $\Delta E' \leftarrow \widetilde{\Delta E}^1 - \widetilde{\Delta E}^2$.
    $W_{\text{new}} \leftarrow \text{Select}(W^1, W^2, \Delta E', \theta)$.
  end if
  $\Delta\text{acc} \leftarrow \text{acc}(W_{\text{new}}) - \text{acc}(W)$.
  $W \leftarrow W_{\text{new}}$.
  $\epsilon \leftarrow \text{Accountant}(\sigma_t, \sigma_v, q_{\text{train}}, q_{\text{val}})$.
  if $\epsilon < \epsilon_B$ **then**
    if $\Delta\text{acc} > p$ **then**
      $(\sigma_t, \sigma_v, \eta) \leftarrow (\sigma_t, \sigma_v, \eta) \cdot \alpha_F$.
    else
      $(\sigma_t, \beta, \eta) \leftarrow (\sigma_t, \beta, \eta) \cdot \alpha_S$.
    end if
  end if
end for
return W.

Definition 17 (Classification accuracy) *Let $\{(x_j, y_j)\}_{j=1}^n$ be a test set with labels y_j and let $\hat{y}_j$ denote the model prediction for input x_j. The classification accuracy is defined as*

$$Acc = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}[\hat{y}_j = y_j], \tag{26}$$

where $\mathbf{1}[\cdot]$ is the indicator function.

Membership inference performance is reported using the area under the ROC curve (AUC), which is standard in the MIA literature[23,24,40–42]. An AUC close to 0.5 corresponds to random guessing and indicates strong privacy protection. In our experiments we use the shadow model based black-box attack and a partial gradient based white-box attack[23,24,40–42]. The same attack code is applied to all methods.

Buffer Size Analysis

We now analyze the impact of buffer size on model accuracy and training efficiency. Figure 4 presents the test accuracy and the final number of iterations when reaching the privacy budget $\epsilon = 1.0$ for different buffer sizes on the MNIST and FMNIST datasets.

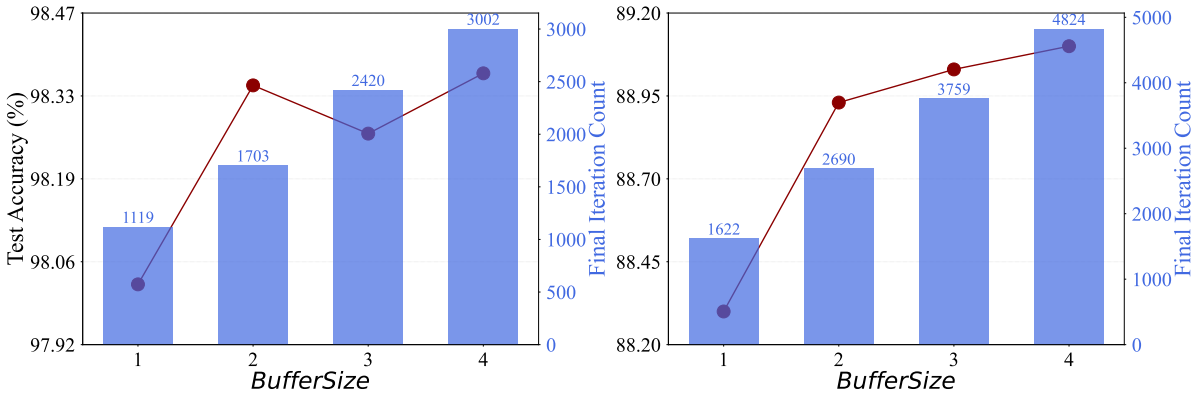

Figure 4. Buffer size analysis: Test accuracy and final iteration count at privacy budget $\epsilon = 1.0$ for different buffer sizes on MNIST and FMNIST.

Clear tradeoffs between model utility and training cost with respect to buffer size can be observed from the figure. The test accuracy of both datasets shows a significant improvement as the buffer size increases from 1 to 2: MNIST rises from approximately 98.0% to 98.35%, and FMNIST increases from 88.3% to 88.95%. This indicates that the dual-candidate buffer mechanism can effectively offset noise interference through local optimization, significantly enhancing model utility. When the buffer size further increases to 3 or 4, the accuracy improvement becomes extremely marginal: MNIST only increases by about 0.05% from Buffer 2 to Buffer 4, and FMNIST by approximately 0.15%, with negligible marginal gains.

The number of iterations grows almost linearly with the buffer size: MNIST increases from 1119 iterations for Buffer 1 to 3002 iterations for Buffer 4, and FMNIST rises from 1622 iterations to 4824 iterations. The training cost of Buffer 4 is nearly twice that of Buffer 2. This is because a larger buffer requires generating and verifying more candidate updates, which significantly increases the overhead of gradient computation and privacy verification.

Setting the buffer size to 2 achieves the optimal balance between accuracy and training efficiency: it not only yields significantly higher accuracy than the single-candidate setting but also avoids the sharp increase in cost and minimal gain when Buffer ≥ 3. Although larger buffers can bring a very slight accuracy improvement, this improvement comes at the cost of several times the training time, resulting in extremely low cost-effectiveness in practical deep learning scenarios, and may even increase the risk of training stagnation due to excessive candidate filtering. Therefore, setting the maximum buffer capacity to 2 can effectively improve update quality through local optimization, control computational overhead, and avoid training stagnation with the decay mechanism, making it the optimal choice that balances model utility, computational efficiency, and anti-stagnation effects.

Overall Performance under Fixed Privacy Budgets

Table 2 reports test accuracy on all four datasets for $\epsilon \in \{1,2,3,4\}$ with $\delta = 10^{-5}$. DPSGD-BR consistently achieves the highest accuracy across all settings. The gains over DPSGD and DPSUR are stable rather than occasional, which suggests that the method is not over-tuned to a single task.

Under the strict privacy budget $\epsilon = 1$, the advantage is already visible. On FMNIST, DPSGD-BR reaches 88.93% accuracy, whereas DPSUR achieves 88.29% and DP-SGD is clearly behind. On CIFAR-10, the gap is more pronounced: DPSGD-BR obtains 65.63% compared to 63.28% for DPSUR and much lower values for DP-SGD and its variants. These settings correspond to high noise regimes where single noisy decisions in DPSUR are particularly unstable. The buffered rejection mechanism appears to provide more reliable update choices and turns the same noise budget into better utility. As the privacy budget becomes less strict, DPSGD-BR maintains its lead. On MNIST and FMNIST, the curves for DPSGD-BR stay near the non-private upper bound and narrow the utility gap associated with DP training. A particularly interesting case arises on CIFAR-10 with $\epsilon = 4$. Here DPSGD-BR slightly surpasses the non-private baseline (72.30% vs 71.12%). This matches the observation that well-calibrated noise can sometimes regularize training and avoid overfitting, even when privacy is not the primary concern.

Overall, the table shows that buffered rejection combined with phased parameter decay yields improvements that are modest on easy tasks and tight budgets, and more substantial on harder tasks and higher noise levels.

Convergence and Privacy Budget Efficiency

We now examine how DPSGD-BR uses the privacy budget over the course of training. Figure 5 compares the number of accepted updates for different methods at the point where $\epsilon = 1.0$ is reached on CIFAR-10 and IMDb. DPSGD-BR attains

Table 2. Test accuracy comparison under different privacy budgets ($\delta = 10^{-5}$)

Dataset	Method	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 4$	Non-private
MNIST	**DPSGD-BR**	**98.39%**	**98.88%**	**98.93%**	**99.15%**	99.11%
	DPSUR[12]	97.95%	98.80%	98.86%	98.95%	
	DPSGD-HF[39]	97.73%	98.40%	98.46%	98.53%	
	DPSGD-TS[21]	97.02%	97.82%	98.12%	98.21%	
	DPSGD[11]	95.13%	96.37%	96.83%	97.22%	
FMNIST	**DPSGD-BR**	**88.93%**	**89.85%**	**90.01%**	**90.44%**	90.98%
	DPSUR[12]	88.29%	89.42%	89.72%	90.10%	
	DPSGD-HF[39]	85.77%	88.44%	88.56%	88.76%	
	DPSGD-TS[21]	78.70%	84.67%	86.14%	86.84%	
	DPSGD[11]	80.24%	82.70%	84.56%	85.39%	
CIFAR-10	**DPSGD-BR**	**65.63%**	**69.96%**	**71.52%**	**72.30%**	71.12%
	DPSUR[12]	63.28%	69.37%	70.59%	72.02%	
	DPSGD-HF[39]	61.65%	66.29%	69.40%	70.53%	
	DPSGD-TS[21]	51.42%	55.18%	61.56%	63.13%	
	DPSGD[11]	47.77%	52.89%	54.79%	58.98%	
IMDb	**DPSGD-BR**	**66.91%**	**72.46%**	**73.53%**	**75.19%**	79.97%
	DPSUR[12]	65.72%	69.57%	73.09%	73.90%	
	DPSGD-TS[21]	65.54%	68.86%	70.02%	70.78%	
	DPSGD[11]	64.36%	68.89%	71.12%	71.59%	

the largest number of effective iterations among all selective update methods. This means that, at the same privacy budget, DPSGD-BR manages to apply more updates that are deemed useful under the noisy validation filter.

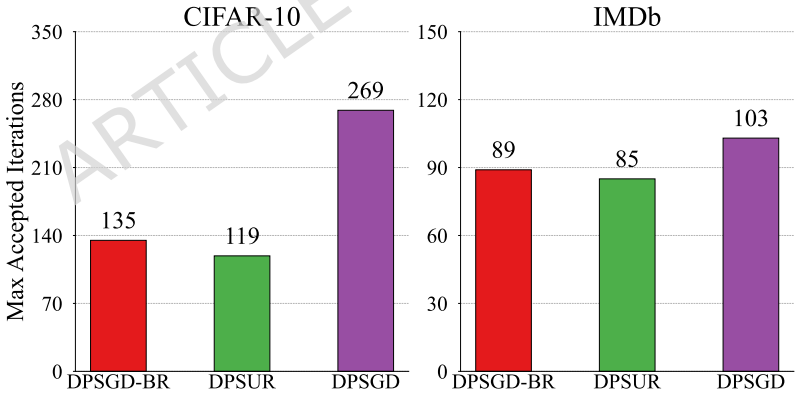

Figure 5. Training efficiency comparison on CIFAR-10 and IMDB: Number of iterations completed when reaching privacy budget $\epsilon = 1.0$.

The convergence curves in Figure 6 plot test accuracy as a function of privacy budget ϵ for all datasets. The DPSGD-BR curves rise faster at the beginning, especially on CIFAR-10 and IMDb. This suggests that the buffered rejection mechanism quickly finds advantageous directions despite heavy noise. The curves are also smoother, with fewer sharp oscillations than DP-SGD baselines. The combination of buffered selection and phased decay appears to stabilize the trajectory on the noisy loss landscape.

From a privacy-budget perspective, these plots show that DPSGD-BR gains more accuracy early, when the available budget is small. The method uses early-stage updates to move the model into a good region, then relies on slower decays and relaxed thresholds to refine performance. This staged behavior aligns with the design of the phased parameter decay strategy and provides empirical support for it.

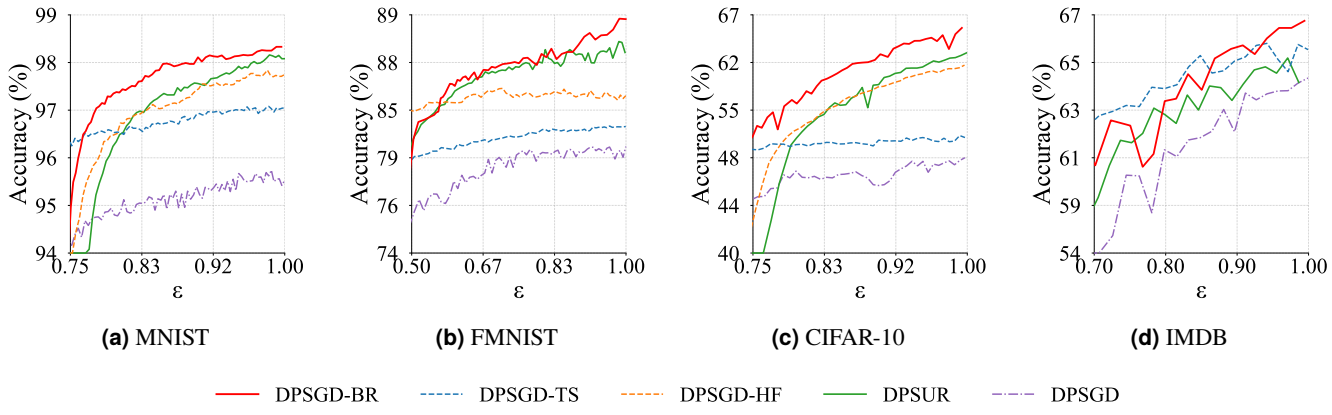

Figure 6. Convergence Curves: Accuracy vs. Privacy Budget (ϵ).

Ablation of Buffer vs. Decay

To understand the contribution of each component, we conduct an ablation study with three variants: only buffering (buffer-on, decay-off), only decay (buffer-off, decay-on), and the full DPSGD-BR (both on). The results are summarized in Figure 7.

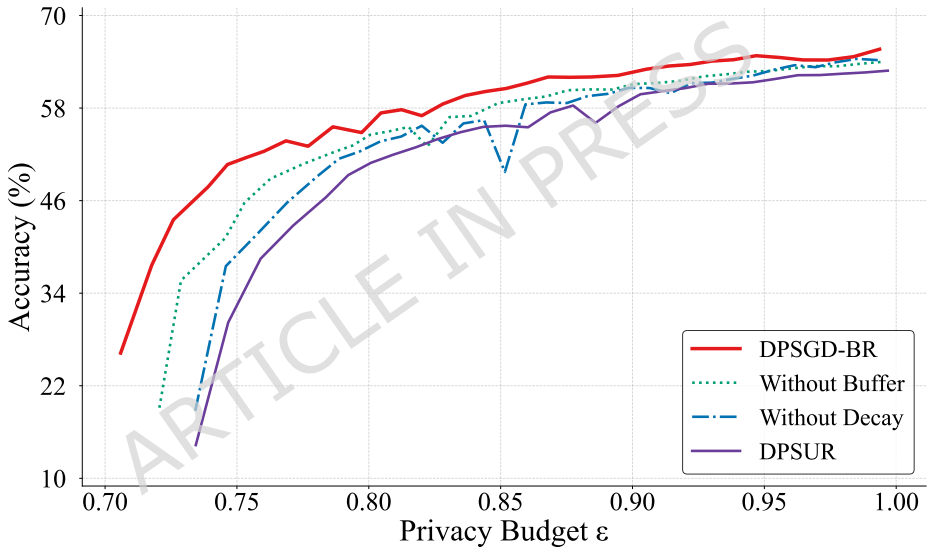

Figure 7. Ablation Study of Buffer and Decay Mechanisms

When only the buffering mechanism is used, the algorithm already improves over DPSUR in the later part of training. The convergence curve is less noisy and reaches higher final accuracy, but the early stage is still somewhat unstable because the noise and learning rate remain large. This variant shows that buffering alone can upgrade single noisy decisions to more robust comparisons, which raises the ceiling on achievable utility. When only the phased decay strategy is used, the curves are smooth and often above DPSUR throughout training. The algorithm benefits from better calibrated noise and learning rate schedules, even without buffered selection. However, the peak accuracy is still below the full DPSGD-BR model. This variant indicates that adaptive decay can reshape the noise regime and give the rejection mechanism a better operating range.

The full DPSGD-BR combines both ideas. Its curve rises quickly, stays stable, and ends with the best accuracy. The ablation study therefore suggests that buffering and decay are complementary: buffering focuses on more reliable update choices at a fixed noise level, while decay improves the noise regime and step size in which these choices are made.

Robustness to Membership Inference Attacks

A natural concern is whether the utility improvements of DPSGD-BR come at the cost of weaker protection against membership inference. To address this, we evaluate black-box and white-box MIA on CIFAR-10. The AUC results are shown in Table 3. Where AUC refers to the Area Under the ROC Curve for membership inference attack evaluation, and a value near 0.5 indicates

the attack behaves close to random guessing.Across all privacy budgets and attack types, DPSGD-BR achieves AUC values close to 0.5. The numbers are similar to or slightly better than those of DP-SGD and DPSUR. In contrast, the non-private models exhibit much higher AUC, indicating strong vulnerability to membership inference. This confirms that the privacy accounting derived under the RDP framework is tight and that the algorithm does not open an obvious side channel through its selective update process.

Table 3. CIFAR-10 Membership Inference Attack (MIA) AUC Values

Attack	Algorithm	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 4$	Non-private
BlackBox	**DPSGD-BR**	**0.501**	**0.500**	**0.502**	**0.506**	**0.732**
Shadow	DPSUR[12]	0.501	0.502	0.503	0.508	
WhiteBox	**DPSGD-BR**	**0.510**	**0.511**	**0.510**	**0.511**	**0.743**
Partial	DPSGD[11]	0.505	0.508	0.510	0.512	

Taken together, these results show that DPSGD-BR improves predictive performance without relaxing the underlying DP guarantee. The buffered rejection and phased decay strategies change how the privacy budget is used but do not weaken the formal protection against MIA.

Hyperparameter Sensitivity

We finally examine the sensitivity of DPSGD-BR to its main hyperparameters on FMNIST with $\epsilon = 1.0$. Figure 8 reports test accuracy as a function of three parameters: the maximum number of consecutive rejections $T_{\max}$, the decay trigger threshold p on Δacc, and the scaling factor k that controls the difference threshold θ in the buffer comparison.

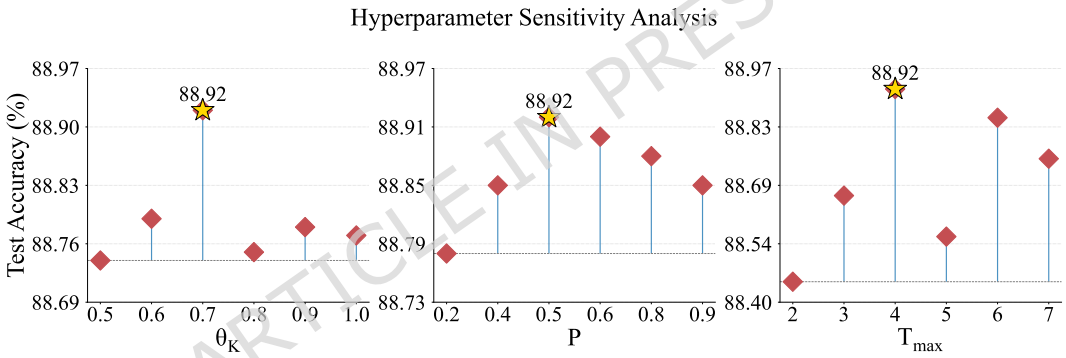

Figure 8. DPSGD-BR Hyperparameter Sensitivity Analysis on FMNIST ($\epsilon = 1.0$)

Table 4. Initial Parameters for DPSGD-BR ($\epsilon = 1$)

Parameter	**MNIST**	**FMNIST**	**CIFAR-10**	**IMDB**
Training Noise (σ_t)	3.0	6.0	14.0	2.3
Learning Rate (η)	2.75	6.0	4.8	0.03
Training Batch Size (batch size)	1024	2048	8192	1024
Validation Clipping Bound (C_v)	0.001	0.001	0.001	0.001
Rejection Threshold (β)	-1.5	-1.5	-1.5	-1.5
Validation Noise (σ_v)	1.37	1.3	1.4	1.37

The curves show that DPSGD-BR is reasonably stable in a broad range around the chosen defaults. For $T_{\max}$ between 3 and 7, accuracy remains close to its peak and degrades only when the threshold is too small or too large. The decay trigger p yields best results around 0.5, but the method is not overly sensitive as long as p stays within a moderate interval. The parameter k also exhibits a wide plateau where performance is nearly flat.

From a practical viewpoint, these plots suggest that DPSGD-BR does not require exhaustive hyperparameter search. A coarse tuning around the reported values is sufficient to obtain strong performance on new tasks, which is important for deployment in privacy-sensitive applications.

Discussion

The experiments jointly support the main design choices behind DPSGD-BR. The buffered rejection mechanism turns the single noisy decision of DPSUR into a local comparison between two candidates. This simple change already leads to better update selection and smoother trajectories under heavy noise. The phased parameter decay strategy then adapts the noise multipliers, thresholds, and learning rate to the current learning stage. Together, these components allow the model to spend its privacy budget on updates that are likely to matter.

Compared with the broader DP-SGD family[11,12,21,39], DPSGD-BR offers a different perspective on efficiency. Rather than focusing only on gradient clipping, learning rate schedules, or optimizer variants, it treats update acceptance itself as a decision process that can be improved under DP. The method keeps the core pipeline of DP-SGD but inserts a lightweight, privacy-aware selection layer and a simple two-phase schedule.

There are also boundaries to where DPSGD-BR may be most effective. The current experiments focus on single-node training with moderate model sizes. In very large-scale or highly distributed settings, such as federated learning, the cost of generating and validating two candidates per accepted update may need to be revisited. On very simple tasks where DP noise is small, the gains over tuned DP-SGD may also be modest. Despite these caveats, the results on MNIST, FMNIST, CIFAR-10, and IMDb indicate that buffered rejection with phased decay is a promising direction for improving DP training in many practical deep learning scenarios.

Conclusion and Future Work

This paper presented DPSGD-BR, a differentially private optimization algorithm that combines a buffered rejection mechanism with a phased parameter decay strategy. The method keeps the basic DP-SGD pipeline but replaces single noisy decisions with a local comparison between two candidates and adapts key training parameters to the current learning stage. This design aims to stabilize updates in high-noise regimes and to use the privacy budget on iterations that are more likely to improve the model. Experiments on MNIST, FMNIST, CIFAR-10, and IMDb show that DPSGD-BR consistently improves test accuracy and convergence behavior under a wide range of privacy budgets. The method achieves higher utility than DP-SGD and DPSUR while maintaining comparable protection against membership inference attacks. The results indicate that buffered rejection and phased decay together provide a practical way to turn the same (ϵ, δ) guarantee into more informative updates. Future work will explore adaptive control of the buffer and thresholds, and investigate how DPSGD-BR extends to large scale or federated learning settings.

Data Availability

This study utilized four publicly available datasets that are openly accessible for research purposes. The MNIST dataset[35] can be accessed at https://github.com/cvdfoundation/mnist/tree/master, Fashion-MNIST[36] at https://github.com/zalandoresearch/fashion-mnist, CIFAR-10[37] at https://www.cs.toronto.edu/~kriz/cifar.html, and the IMDb movie reviews dataset[38] at http://ai.stanford.edu/~amaas/data/sentiment/. All datasets are freely available and can be obtained from these official repositories without restrictions.

References

1. Hong, C., Chen, L., Liang, Y. & Zeng, Z. Stacked capsule graph autoencoders for geometry-aware 3d head pose estimation. *Comput. Vis. Image Underst.* **208**, 103224 (2021).
2. Huang, J., Hong, C., Xie, R., Ran, L. & Qian, J. A simple and efficient channel mlp on token for human pose estimation. *Int. J. Mach. Learn. Cybern.* **16**, 3809–3817 (2025).
3. Lee, X., Hong, C., Zhang, X. & Chen, Y. Droformer: temporal action detection with drop mechanism of attention. *Int. J. Mach. Learn. Cybern.* 1–16 (2025).
4. Luo, G.-F., Wang, D.-H., Zhang, X.-Y., Lin, Z.-H. & Zhu, S. Joint radical embedding and detection for zero-shot chinese character recognition. *Pattern Recognit.* **161**, 111286 (2025).
5. Luo, G.-F. *et al.* Self-information of radicals: A new clue for zero-shot chinese character recognition. *Pattern Recognit.* **140**, 109598 (2023).
6. Chen, S. *et al.* Learning relationship-guided vision-language transformer for facial attribute recognition. *Pattern Recognit.* **170**, 112063 (2026).
7. Chen, S. *et al.* Hierarchical token-aware cross-modality reconstruction for visible-infrared person re-identification. *IEEE Transactions on Multimed.* (2025).

8. Lian, J., Wang, D.-H., Wu, Y. & Zhu, S. Multi-branch enhanced discriminative network for vehicle re-identification. *IEEE Transactions on Intell. Transp. Syst.* **25**, 1263–1274 (2023).

9. Dwork, C., Roth, A. *et al.* The algorithmic foundations of differential privacy. *Foundations trends theoretical computer science* **9**, 211–407 (2014).

10. Paszke, A. *et al.* Automatic differentiation in pytorch. *Neural Inf. Process. Syst.* (2017).

11. Abadi, M. *et al.* Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 308–318 (2016).

12. Fu, J. *et al.* Dpsur: Accelerating differentially private stochastic gradient descent using selective update and release. *Proc. VLDB Endow* (2024).

13. Hu, J., Liu, Y. & Wu, K. Neural network pruning based on channel attention mechanism. *Connect. Sci.* **34**, 2201–2218 (2022).

14. Katharopoulos, A. & Fleuret, F. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, 2525–2534 (2018).

15. Phan, N., Wu, X., Hu, H. & Dou, D. Adaptive laplace mechanism: Differential privacy preservation in deep learning. In *2017 IEEE international conference on data mining (ICDM)*, 385–394 (2017).

16. Andrew, G., Thakkar, O., McMahan, B. & Ramaswamy, S. Differentially private learning with adaptive clipping. *Adv. Neural Inf. Process. Syst.* **34**, 17455–17466 (2021).

17. Lee, J. & Kifer, D. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1656–1665 (2018).

18. Pichapati, V., Suresh, A. T., Yu, F. X., Reddi, S. J. & Kumar, S. Adaclip: Adaptive clipping for private sgd. *arXiv preprint arXiv:1908.07643* (2019).

19. Koskela, A. & Honkela, A. Learning rate adaptation for differentially private learning. *Int. Conf. on Artif. Intell. Stat.* 2465–2475 (2020).

20. De, S., Berrada, L., Hayes, J., Smith, S. L. & Balle, B. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650* (2022).

21. Papernot, N., Thakurta, A., Song, S., Chien, S. & Erlingsson, U. Tempered sigmoid activations for deep learning with differential privacy. In *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, 9312–9321 (2021).

22. Bassily, R., Smith, A. & Thakurta, A. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th annual symposium on foundations of computer science*, 464–473 (2014).

23. Carlini, N., Liu, C., Erlingsson, U., Kos, J. & Song, D. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, 267–284 (2019).

24. Fredrikson, M., Jha, S. & Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 1322–1333 (2015).

25. Zhang, K., Yuan, X., Sun, R., Hong, C. & Xue, J. Traceable and collision-resilient differential privacy. *IEEE Transactions on Inf. Forensics Secur.* **20**, 11816–11829 (2025).

26. Zhang, K. *et al.* Dpnm: A differential private notary mechanism for privacy preservation in cross-chain transactions. *IEEE Transactions on Inf. Forensics Secur.* (2025).

27. Mironov, I. Renyi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, 263–275 (2017).

28. Mironov, I., Talwar, K. & Zhang, L. Renyi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv:1908.10530* (2019).

29. Zhang, K. *et al.* Bounded and unbiased composite differential privacy. In *2024 IEEE Symposium on Security and Privacy (SP)*, 972–990 (2024).

30. Zhang, K. *et al.* Towards privacy in decentralized iot: A blockchain-based dual response dp mechanism. *Big Data Min. Anal.* **7**, 699–717 (2024).

31. Dwork, C., McSherry, F., Nissim, K. & Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, 265–284 (2006).

32. Balle, B. & Wang, Y.-X. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International conference on machine learning*, 394–403 (2018).

33. Cummings, R. & Desai, D. The role of differential privacy in gdpr compliance. In *FAT'18: Proceedings of the Conference on Fairness, Accountability, and Transparency*, vol. 20, 2 (2018).

34. Bu, Z., Dong, J., Long, Q. & Su, W. J. Deep learning with gaussian differential privacy. *Harv. data science review* **2020**, 10–1162 (2020).

35. Deng, L. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine* **29**, 141–142 (2012).

36. Xiao, H., Rasul, K. & Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

37. Alex, K. Learning multiple layers of features from tiny images. *learning-features-2009-TR* (2009).

38. Maas, A. *et al.* Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 142–150 (2011).

39. Tramer, F. & Boneh, D. Differentially private learning needs better features (or much more data). *Int. Conf. on Learn. Represent.* (2020).

40. Nasr, M., Shokri, R. & Houmansadr, A. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, 739–753 (2019).

41. Melis, L., Song, C., De Cristofaro, E. & Shmatikov, V. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, 691–706 (2019).

42. Jayaraman, B. & Evans, D. Evaluating differentially private machine learning in practice. In *28th USENIX security symposium (USENIX security 19)*, 1895–1912 (2019).

Funding

This work was supported by the Natural Science Foundation of Xiamen, China (Grant No. 3502Z202472027); the Natural Science Foundation of Fujian Province, China (Grant No. 2025J011277); the Educational Research Projects for Young and Middle-aged Teachers of Fujian Province, China (Grant No. JAT241118); the Xiamen Municipal Research Program for Returned Overseas Scholars (Grant No. XMHRSS-[2024]-241-03), under the administration of the Xiamen Municipal Human Resources and Social Security Bureau, China; and the Research Start-up Program for High-level Talents at Xiamen University of Technology (Grant No. YKJ24006R).

Author Contributions

S.D. conceived the study, developed the methodology, implemented the software, performed validation experiments, and wrote the initial draft of the manuscript. K.Z. was responsible for data curation, contributed to methodology development, created visualizations, and provided supervision throughout the project. W.Z. contributed to methodology development and managed project administration. H.J. contributed to the development of methodology. P.-W.T. supervised the research and critically revised the manuscript. All authors reviewed and approved the final version of the manuscript for submission.

Competing Interests

The author(s) declare no competing interests.