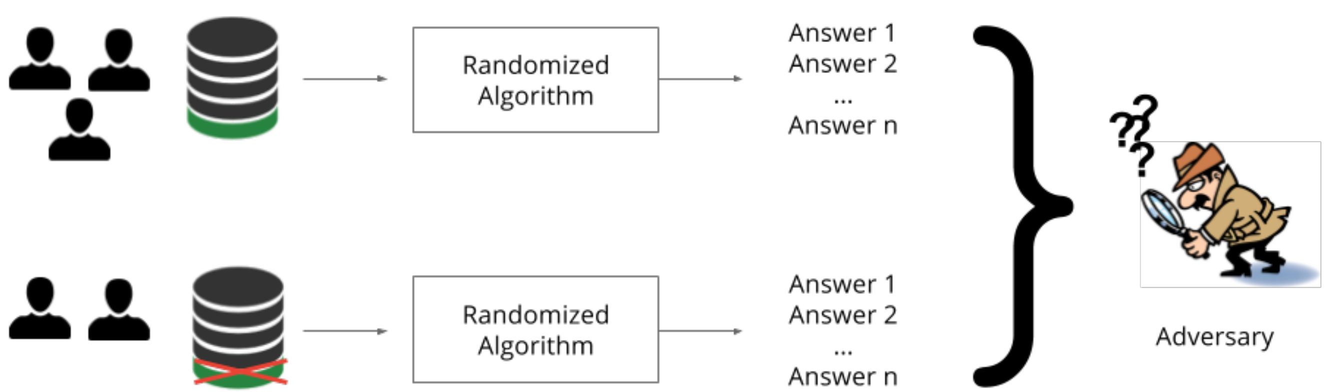
Privacy-Preserving
Gradient Boosting Decision Trees

Kobe University Engineering ES5
M2 201T264T
Fuki Yamamoto

Background

✓ Recently, privacy concerns are growing
(e.g., inference of original data from statistics, leakage of sensitive info etc.)
→ Related legislation (e.g. General Data Protection Regulation (GDPR))
→ Balancing privacy protection and data utilization is required
→ Privacy-Preserving Data Mining

✓ Differential privacy (DP) provides privacy-preserving statistics
(Other privacy-preserving techniques protect only the computational process)



Background

✓ Previous DP algorithms in GBDT suffer from serious accuracy loss (GBDT is a ML model used in various domain)

✓ In this study, the accuracy loss is improved by focusing on **bounding of sensitivity** and **privacy budget allocation**

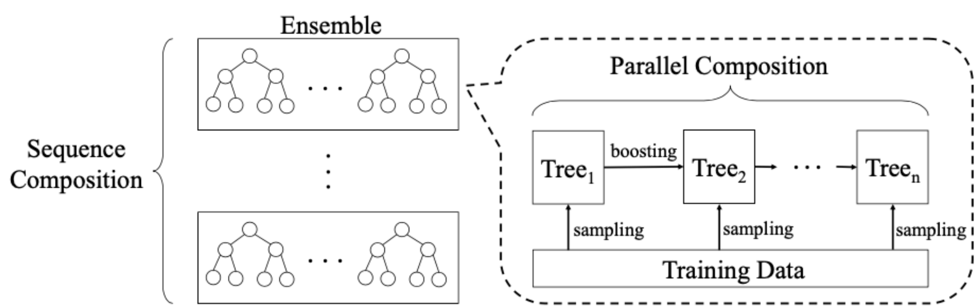
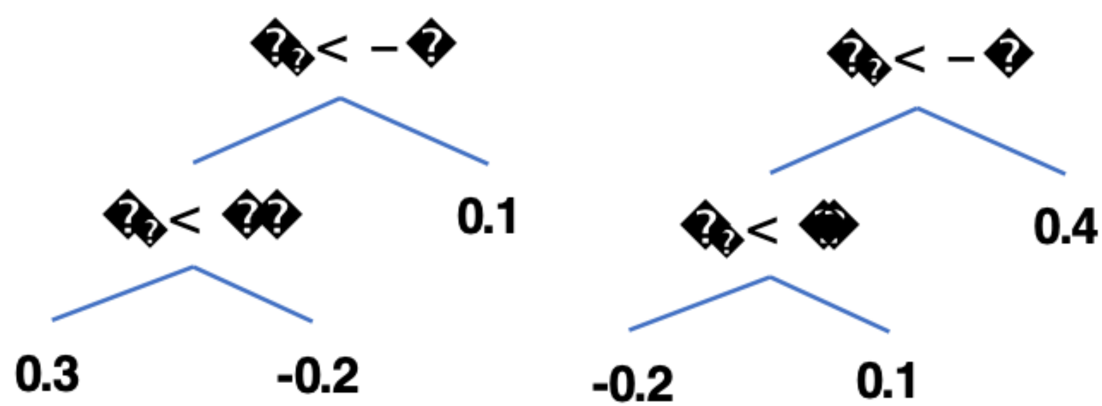


Figure 1: The two-level boosting design of DPBoost

Gradient Boosting Decision Trees (GBDT)

- ✓ Ensemble model with decision tree as the base model
- ✓ Training by gradient boosting, which improves the cost function by sequentially building the base model
- ✓ High efficiency and predictive performance, useful in various fields



Red: $0.3 + 0.4 = 0.7 > 0 \Longrightarrow$ **Yes**

Purple: $0.1 - 0.2 = -0.1 < 0 \Longrightarrow$ **No**

GBDT: Training

Cost function (for t-th tree)

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \left(g_i f_t(x_i) + \frac{1}{2} f_t^2(x_i) \right) + \Omega(f_t)$$

Leaf Weight

$$V(I) = -\frac{\sum_{i \in I} g_i}{|I| + \lambda}$$

Splitting

$$G(I_L, I_R) = \frac{\left(\sum_{i \in I_L} g_i\right)^2}{|I_L| + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{|I_R| + \lambda}$$

I_L, I_R: instance sets of left and right nodes
I: $I_L \cup I_R$

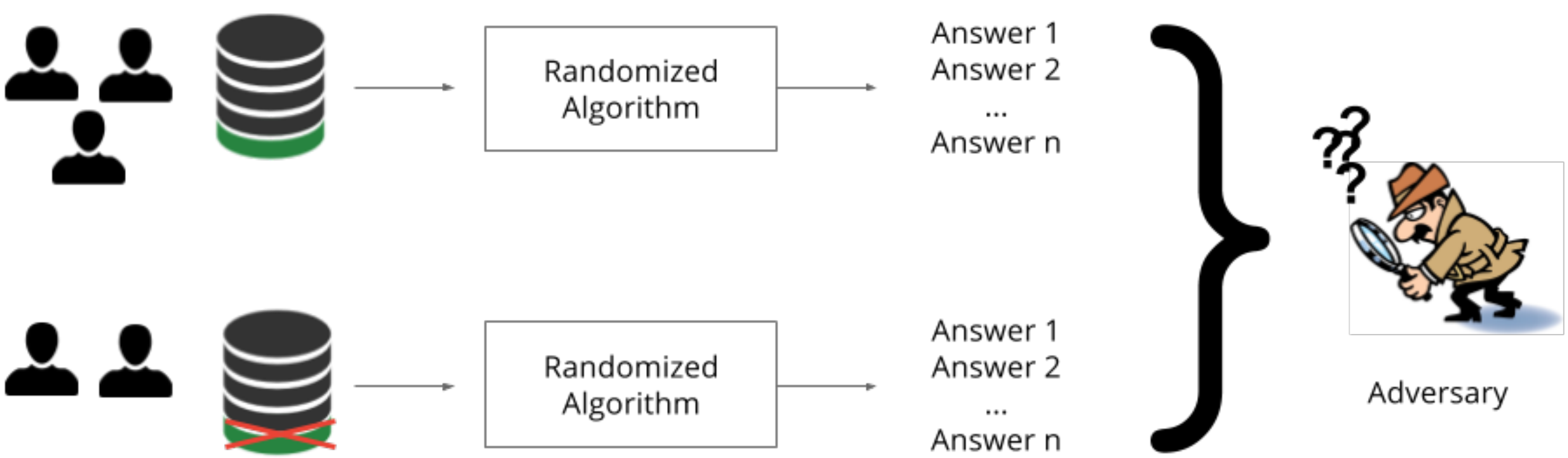
g_i: $\partial_{\hat{y}} l(y_i, \hat{y})$
l: loss function (square error)
$\hat{y}$: prediction by previous trees

$f_t(\cdot)$: output of t-th tree
$\Omega(\cdot)$: regularization term
λ: regularization parameter

Differential Privacy (DP)

- ✓ Powerful and quantitative **privacy-preserving technology**
- ✓ An attacker with arbitrary background knowledge

  (i.e., information about all the data contained in the dataset) is assumed
- ✓ Random noise is added to the statistics to obscure individual data



DP: Formula

- ✓ Adding random noise (F) makes it difficult to discriminate between neighboring datasets ($D_1,\ D_2 \in D$)
- ✓ The smaller ϵ is, the stronger the privacy is protected
- ✓ Effective also against an attacker with information on D

ϵ-Differential Privacy For $D_1, D_2 \in D$,

$$\Pr\left[F\left(D_1\right) \in O\right] \leq e^{\epsilon} \cdot \Pr[F\left(D_2\right) \in O]$$

F: randomized function, ϵ: privacy budget,
O: any output of f, D: entire dataset,

D_1, D_2: datasets that differ in a single record ($D_1, D_2 \in D$)

DP: Sensitivity

✓ Dataset dependency of the output defined for each f

✓ The lower sensitivity, the smaller the required noise for DP

$$\boxed{\Delta f = \max_{D_1, D_2 \in D} \left\| f(D_1) - f(D_2) \right\|_1}$$

Δf: sensitivity for f, f: original function, D: entire dataset,
D_1, D_2: datasets that differ in a single record (D_1, $D_2 \in D$)

DP: Mechanisms

Laplace Mechanism: (for a query of **continuous** values)

Add a random number generated from the Laplace distribution

$$F(D) = f(D) + Lap(0, \frac{\Delta f}{\epsilon})$$

Exponential Mechanism: (for a query of **discrete** values)

Convert to a probabilistic algorithm based on a gain function

$$F(D, u) = \text{choose } r \in R \text{ with probability} \propto \exp(\frac{\epsilon u(D, r)}{2\Delta u})$$

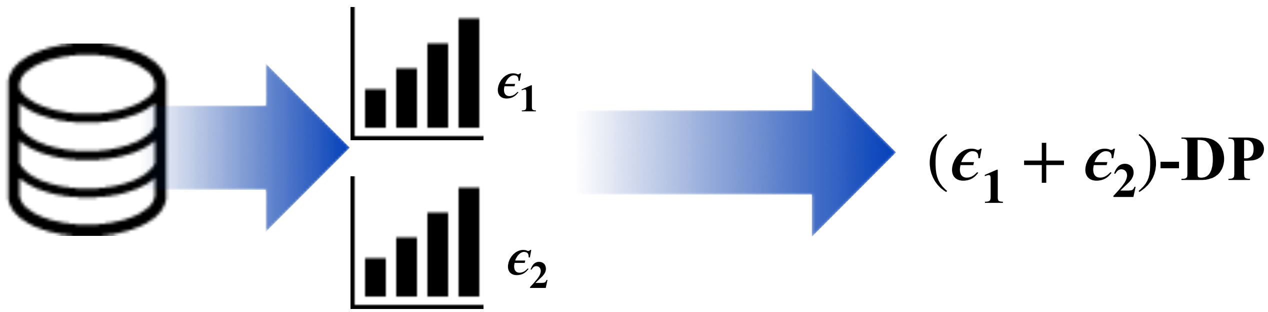
F: mechanisms, f: original function, ϵ: privacy budget,
Lap: Laplace distribution, u: gain function $(D \times R) \rightarrow \mathbb{R}$

DP: Composition

Sequential Composition

(A series of functions $f = \{f_1, \ldots, f_m\}$ is executed on the **same dataset**)

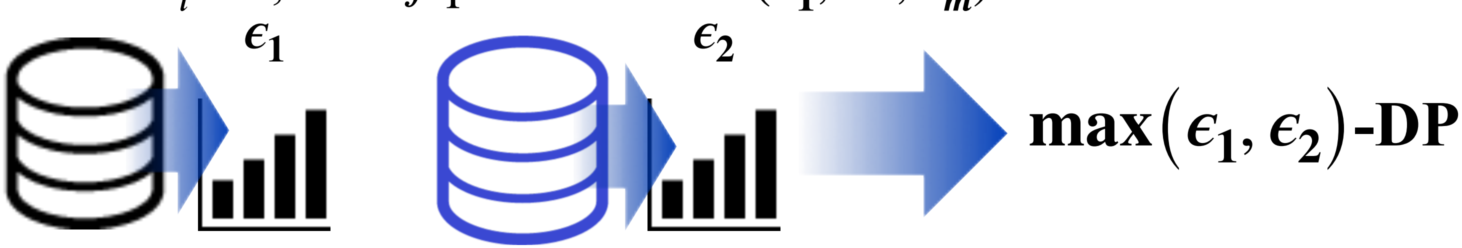
If f_i provides ϵ_i-DP, then f provides $\boldsymbol{\sum_{i=1}^{m} f_i}$**-DP**



Parallel Composition

(A series of functions $f = \{f_1, \ldots, f_m\}$ is performed on **disjoint datasets**)

If f_i provides ϵ_i-DP, then f provides $\mathbf{max}(\epsilon_1, \ldots, \epsilon_m)$**-DP**



Related Works

Liu et al.'s method

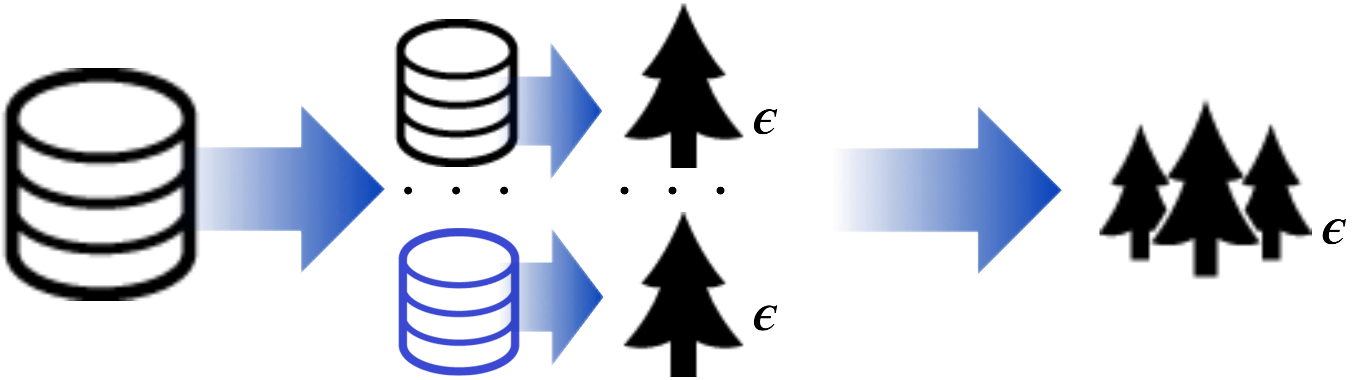
Simply apply sequential composition

→ Scale of noise increases in proportion to the number of trees

Zhao et al's method

Simply apply parallel composition

→ Data samples for each tree decrease in proportion to the number of trees



Abstract

Title: Privacy-Preserving Gradient Boosting Decision Trees

proposed **DPBoost**, a DP framework that improves the accuracy loss in GBDT

Contributions:

✓ **Sensitivity Bounds**

- More tightly and accurately bounded sensitivity
- Gradient-based Data Filtering (GDF)
- Geometric Leaf Clipping (GLC)

✓ **Privacy Budget Allocations**

- Ensemble of Ensembles (EoE)

Tighter Sensitivity Bounds

Conventionally,

sensitivity was bounded by estimating the output range of the function

$$\Delta G \leq 2\max\left|\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\left|I_L\right| + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\left|I_R\right| + \lambda}\right|, \Delta V \leq 2\max\left|\frac{\sum_{i\in I} g_i}{\left|I\right| + \lambda}\right|$$

→ **Proportional** to the number of data samples

DPBoost

bound sensitivity more accurately based on definition

$$\Delta f = \max_{D_1, D_2 \in D} \left\| f\left(D_1\right) - f\left(D_2\right) \right\|_1$$

$$\Longrightarrow \Delta G \leq \frac{3\lambda + 2}{(\lambda + 1)(\lambda + 2)} g^{*2}, \Delta V \leq \frac{g^*}{1 + \lambda} \quad (g^* = \max_{i\in D}\left|g_i\right|)$$

→ **Not proportional** to the number of data samples

Gradient-based Data Filtering (GDF)

✓ Sensitivity can be bounded more tightly by filtering out data with large g_i

→ Filter out data with gradients larger than g_l^*

$$g_l^* = \max_{y_p \in [-1,\, 1]} \left\| \frac{\partial l\left(y_p, y\right)}{\partial y} \Big|_{y=0} \right\| \Longrightarrow \Delta G \leq \frac{3\lambda + 2}{(\lambda+1)(\lambda+2)} g_l^{*2},\ \Delta V \leq \frac{g_l^*}{1+\lambda}$$

✓ g_l^* depends only on the loss function l, and $g_l^* = 1$ when $l =$ MSE

✓ Table 3 shows that GDF filters only a small number of data

Table 3: The number of training instances with/without gradient-based data filtering

	w/ GDF	w/o GDF	filtered ratio
abalone	3340	3292	1.44%
YearPredictionMSD	370902	340989	8.06%
sklearn_reg	800000	799999	0

Geometric Leaf Clipping (GLC)

✓ Sensitivity can be bounded more tightly by clipping leaf weight $V(I)$

→ However, simple clipping causes a severe accuracy loss

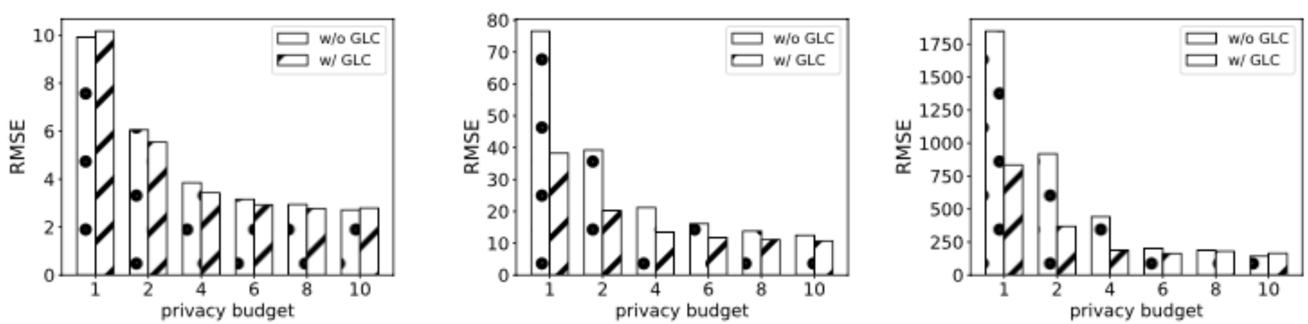
→ Clipping according to gradient decay in GBDT learning

→ However, deriving the exact decay pattern is difficult

✓ When only one data corresponds to each leaf,

$$|V_t| \leq g_l^*(1-\eta)^{t-1} \Longrightarrow \Delta V \leq \min\left(\frac{g_l^*}{1+\lambda}, 2g_l^*(1-\eta)^{t-1}\right)$$

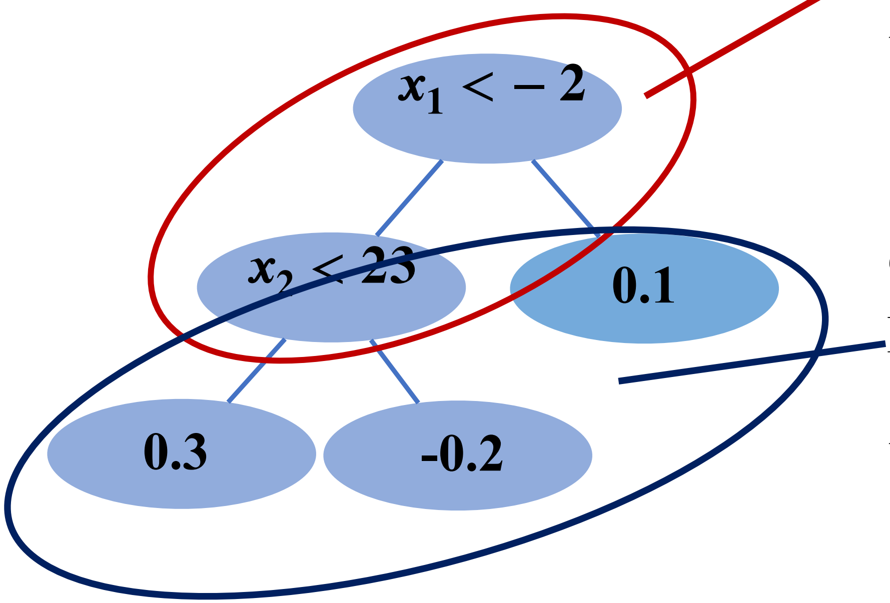
✓ The following figure shows that GLC leads to improved accuracy



(a) abalone (b) YearPrediction (c) synthetic_reg

Privacy Budget Allocations for A Single Tree

- ✓ Introducing methods similar to those used in previous studies
- ✓ Applying exponential mechanism for splitting,
  Laplace mechanism for leaf weight
- ✓ Allocating ε/2 to splitting and ε/2 to leaf weights



Exponential Mechanism

$F(D)$ = choose split points with

$$\text{probability} \propto \exp\left(\frac{\epsilon G\left(I_L, I_R\right)}{2\Delta G}\right)$$

G: split score, $\Delta G \leq \frac{3\lambda + 2}{(\lambda + 1)(\lambda + 2)} g_l^{*2}$

Laplace Mechanism

$$F(D) = V(D) + Lap\left(0, \frac{\Delta V}{\epsilon}\right)$$

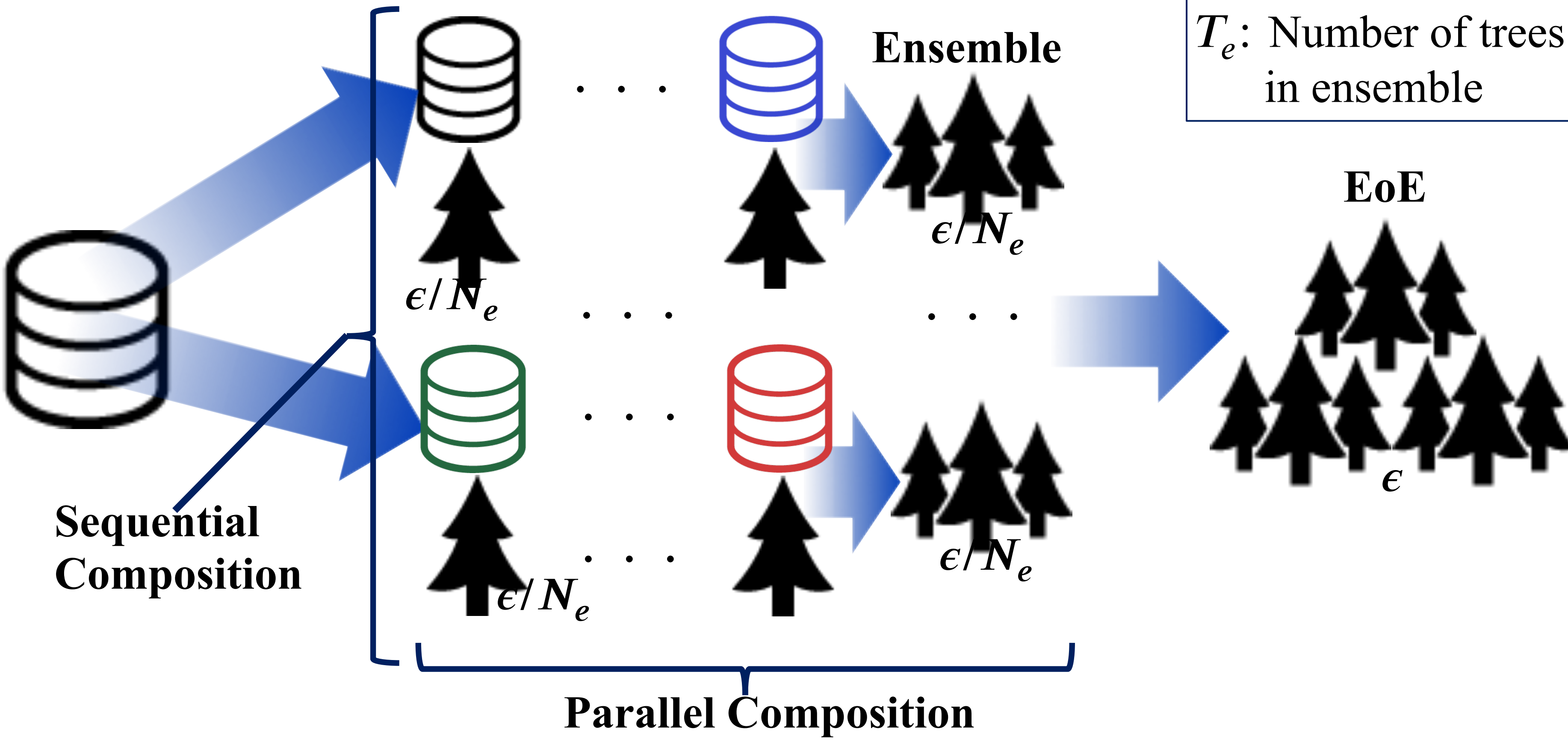
V: leaf weight

$$\Delta V: \ \leq \min\left(\frac{g_l^*}{1 + \lambda}, 2g_l^*\left(1 - \eta\right)^{t-1}\right)$$

Privacy Budget Allocations Across Trees

- ✓ Applying parallel composition prevents the reduction of noise
- ✓ Ensemble of ensembles (EoE) reduces the accuracy loss caused by parallel composition

N_e: Number of ensembles,
T_e: Number of trees in ensemble



Privacy Budget Allocations Across Trees

- ✓ Applying parallel composition prevents the reduction of noise
- ✓ Ensemble of ensembles (EoE) reduces the accuracy loss

  caused by parallel composition

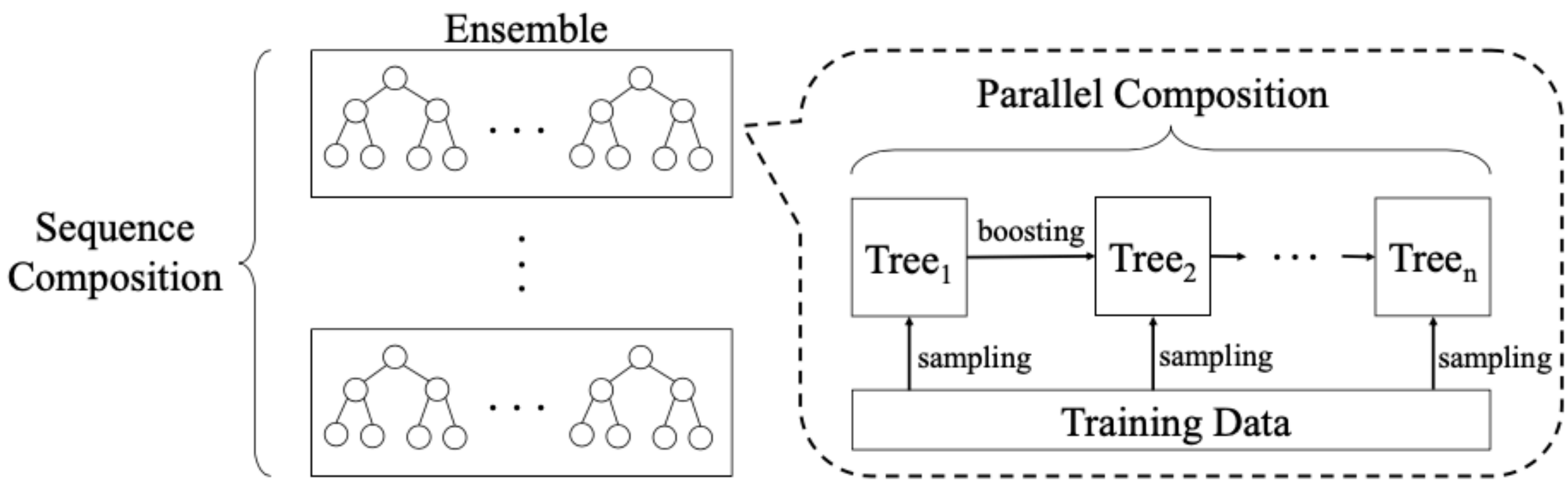


Figure 1: The two-level boosting design of DPBoost

Experimental Setups

- ✓ Use following public datasets
- ✓ Compare DPBoost with the following three methods:
  - **PARA** (Zhao et al.'s approach, 2018)
  - **SEQ** (Liu et al.'s approach, 2018)
  - **NP** (Train GBDTs without privacy concerns)
- ✓ Tighter sensitivity (proposed in this study) is applied to all methods

Table 1: Datasets used in the experiments.

datasets	#data	#features	task
adult	32,561	123	classification
real-sim	72,309	20,958	
covtype	581,012	54	
susy	5,000,000	18	
cod-rna	59,535	8	
webdata	49,749	300	
synthetic_cls	1,000,000	400	
abalone	4,177	8	regression
YearPredictionMSD	463,715	90	
synthetic_reg	1,000,000	400	

Experiments: Accuracy for different ϵ

- ✓ Set N_e to 1 in DPBoost and T to 50 for all approaches
  → Validation of **effects of GDF, GLC**
- ✓ DPBoost always outperformed the previous studies in small ϵ

N_e: Number of ensembles,
T: Total number of trees

N_e: number of ensembles, T: total number of trees

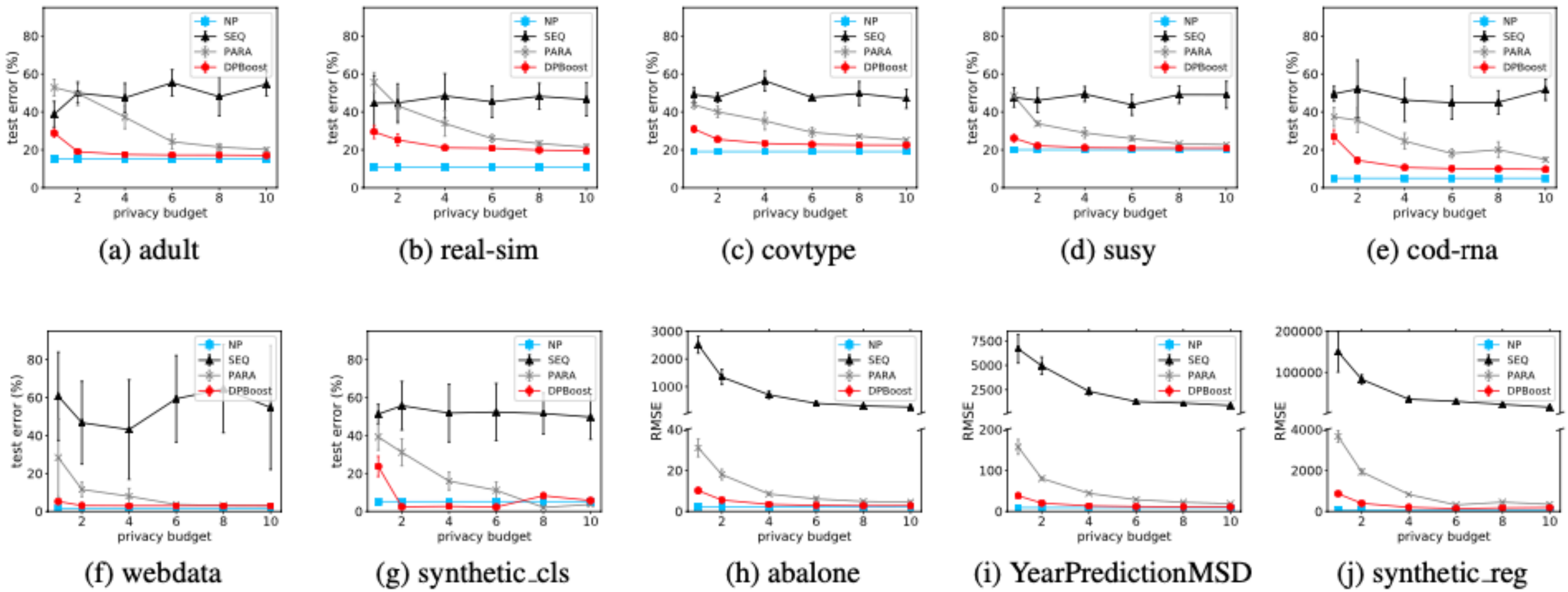
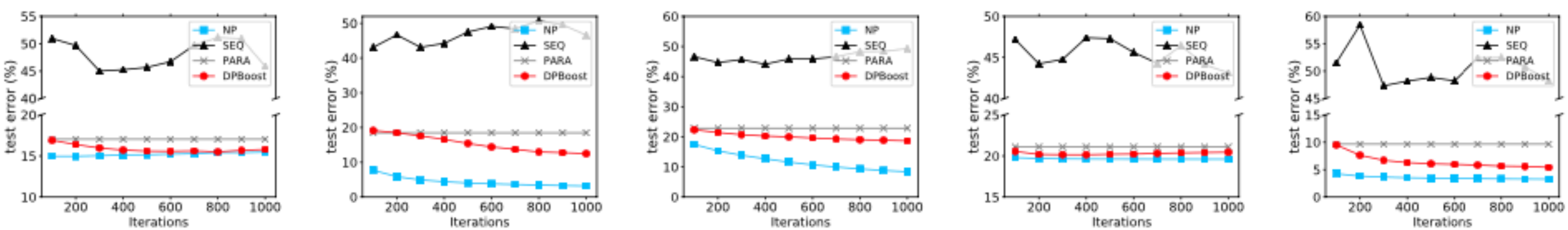


Figure 2: Comparison of the test errors/RMSE given different total privacy budgets. The number of trees is set to 50.

Experiments: Accuracy for different iterations

- ✓ Set N_e to 20 in DPBoost and T to 20 ~ 1000, $\epsilon = 100$ for all approaches
  → Validation of **effects of EoE**
- ✓ Unlike PARA and SEQ, DPBoost's accuracy was improved by increasing T

N_e: Number of ensembles,
T: Total Number of trees



(a) adult (b) real-sim (c) covtype (d) susy (e) cod-rna

Figure 3: Comparison of test error convergence. The number of trees is set to 1000.

Experiments: Training Time

- ✓ Exponential mechanism (split point computation) is costly
- ✓ Time overhead increases in proportion to the size of the dataset

Table 2: Training time per tree (second) of DPBoost and NP.

datasets	DPBoost	NP
adult	0.019	0.007
real-sim	2.97	0.82
covtype	0.085	0.044
SUSY	0.38	0.32
cod-rna	0.016	0.009
webdata	0.032	0.013
synthetic_cls	1.00	0.36
abalone	2.95	2.85
YearPrediction	0.38	0.12
synthetic_reg	0.96	0.36

Conclusions

DPBoost:

✓ bounded sensitivity more strictly (GDF, GLC)

✓ allocated privacy budget more appropriately (EoE)

✓ achieved higher accuracy than previous studies

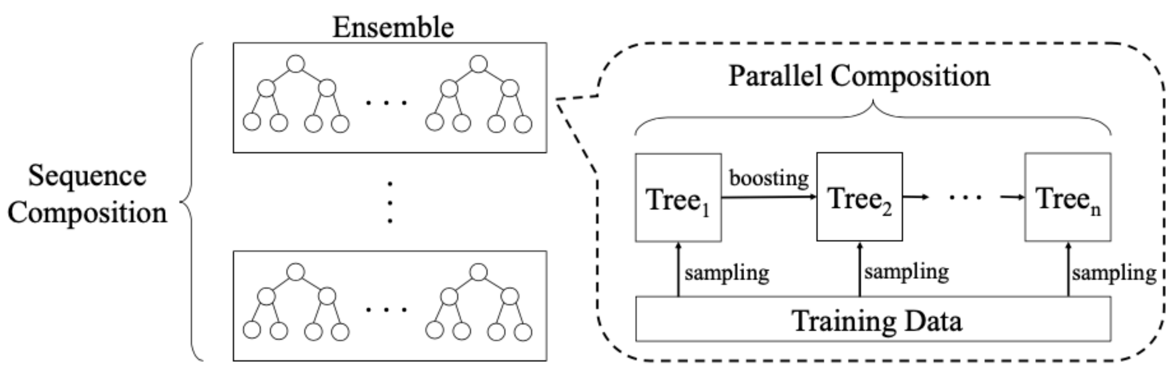


Figure 1: The two-level boosting design of DPBoost

Privacy Budget Allocations for A Single Tree

Algorithm 1: TrainSingleTree: Train a differentially private decision tree

Input: I: training data, $Depth_{max}$: maximum depth
Input: ε_t: privacy budget

$\varepsilon_{leaf} \leftarrow \frac{\varepsilon_t}{2}$ // privacy budget for leaf nodes
$\varepsilon_{nleaf} \leftarrow \frac{\varepsilon_t}{2 Depth_{max}}$ // privacy budget for internal nodes
Perform gradient-based data filtering on dataset I.
for $depth = 1$ **to** $Depth_{max}$ **do**
    for *each node in current depth* **do**
        for *each split value* i **do**
            Compute gain G_i according to Equation (3).
            $P_i \leftarrow exp(\frac{\varepsilon_{nleaf} G_i}{2 \Delta G})$
        /* Apply exponential mechanism */
        Choose a value s with probability $(P_s / \sum_i P_i)$.
        Split current node by feature value s.
for *each leaf node* i **do**
    Compute leaf value V_i according to Equation (4).
    Perform geometric leaf clipping on V_i.
    /* Apply Laplace mechanism */
    $V_i \leftarrow V_i + Lap(0, \Delta V / \varepsilon_{nleaf})$

Output: A ε_t-differentially private decision tree

Privacy Budget Allocations Across Trees

Algorithm 2: Train differentially private GBDTs

Input: $\mathcal{D}$: Dataset, $Depth_{max}$: maximum depth
Input: ε: privacy budget, λ: regularization parameter
Input: T: total number of trees, l: loss function
Input: T_e: number of trees in an ensemble

$N_e \leftarrow \lceil T/T_e \rceil$ // the number of ensembles
$\varepsilon_e \leftarrow \varepsilon/N_e$ // privacy budget for each tree
for $t = 1$ **to** T **do**
  Update gradients of all training instances on loss l.
  $t_e \leftarrow t \mod T_e$
  if $t_e == 1$ **then**
    $I \leftarrow \mathcal{D}$ // initialize the dataset for an ensemble
  Randomly pick $\left(\frac{|\mathcal{D}|\eta(1-\eta)^{t_e}}{1-(1-\eta)^{T_e}}\right)$ instances from I to constitute the subset I_t.
  $I \leftarrow I - I_t$
  TrainSingleTree (dataset = I_t,
    maximum depth = $Depth_{max}$,
    privacy budget = ε_e,
    $\Delta G = \frac{3\lambda+2}{(\lambda+1)(\lambda+2)} g_l^{*2}$,
    $\Delta V = \min(\frac{g_l^*}{1+\lambda}, 2g_l^*(1-\eta)^{t-1})$).

Output: ε-differentially private GBDTs

Applications of DP and GBDT