西安電子科技大學
XIDIAN UNIVERSITY

Kangaroo: A Private and Amortized Inference Framework over WAN for Large-Scale Decision Tree Evaluation

Hui Zhu/Wei Xu

Xidian University

February 25, 2026

Email: xuwei_1@stu.xidian.edu.cn

Wei Xu, Hui Zhu, Yandong Zheng, Song Bian, Ning Sun, Hao Yuan, Dengguo Feng, and Hui Li. Kangaroo: A Private and Amortized Inference Framework over WAN for Large-Scale Decision Tree Evaluation. NDSS 2026.

Outline

- Introduction
- Preliminaries
- Framework Overview
- Building Blocks
- Our Optimizations
- Performance Evaluation
- Conclusion

- **Decision Tree**

Decision Tree[1], which are capable of efficiently handling complex tasks, has been widely applied in various fields such as medical diagnosis, financial risk assessment, and customer behavior prediction.

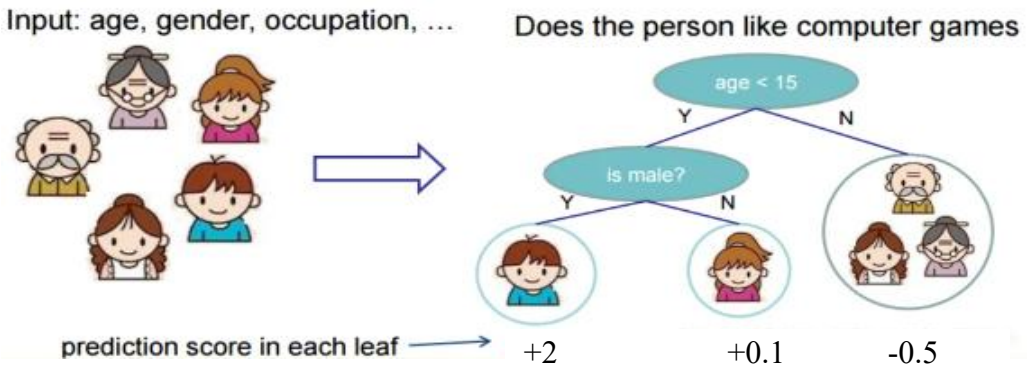


Fig 1. The decision tree

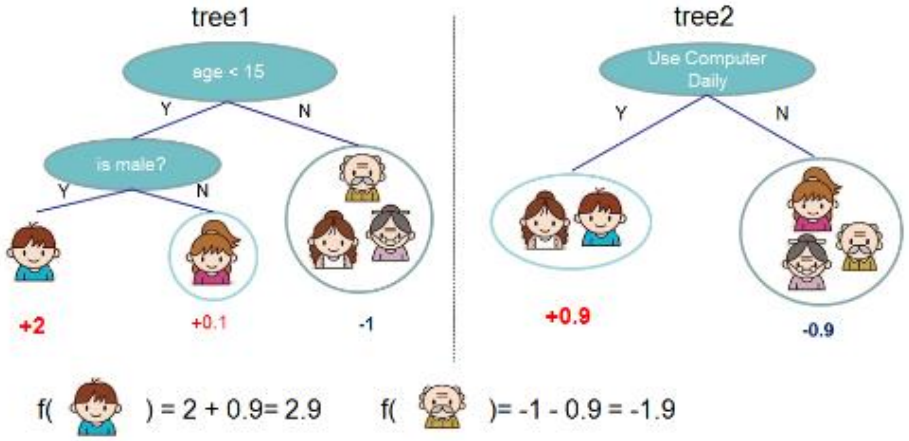


Fig 2. The random forest

Specially, ensemble trees, such as random forest, can combine multiple trees to enhance the accuracy of predictions, which have been widely used in various competitions[2].

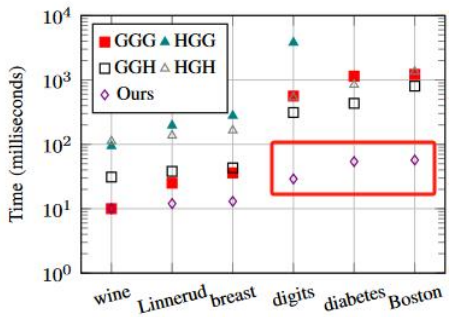
[1] Costa, Vinícius G., and Carlos E. Pedreira. "Recent advances in decision trees: An updated survey." Artificial Intelligence Review 56.5 (2023): 4765-4800.
[2] https://cloud.tencent.com/developer/article/2237712

- **Private Decision Tree Evalution**

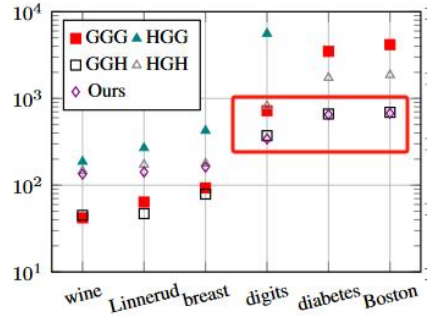
To protect data and model security (User's feature, Evaluation result, True path, Model parameters, etc), many private decision tree evalution schemes (PDTE) have been proposed (Selection, Comparison, Evaluation). Based on the number of communication rounds, these schemes can be categorized into **depth-related evaluation schemes**[1], and **constant-round evaluation schemes**[2,3,4].

- Depth-related evaluation schemes -- Commuication cost O(D), Computation cost O(D)
- Constant-round evaluation schemes -- Commuication cost O(1), Computation cost O(τ) (AHE)

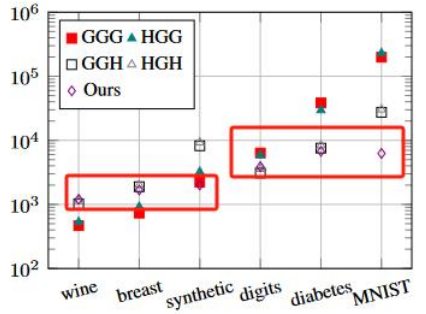
* D is depth，τ is the node number.



(a) UCI datasets on LAN (1Gbps/0.1ms)



(b) UCI datasets on MAN (100Mbps/6ms)



(c) Selected tests on WAN (40Mbps/80ms)

[1] Ma, Jack PK, et al. "Let's stride blindfolded in a forest: Sublinear multi-client decision trees evaluation." NDSS (2021).
[2] Kiss, Ágnes, et al. "SoK: Modular and efficient private decision tree evaluation." Proceedings on Privacy Enhancing Technologies 2 (2019).
[3] Tai, Raymond KH, et al. "Privacy-preserving decision trees evaluation via linear functions." Computer Security–ESORICS 2017.
[4] Akhavan Mahdavi, Rasoul, et al. "Level up: Private non-interactive decision tree evaluation using levelled homomorphic encryption." CCS. 2023.

- **Motivation**

Existing privacy-preserving schemes struggle to scale effectively to support large, deep, and sparse models in real-world deployment environments with high latency and limited bandwidth.

- ➢ Depth-related evaluation schemes -- Huge commuication cost, influenced by tree depth
- ➢ Constant-round evaluation schemes -- Huge commuication and computation cost, influenced by tree node number

The main motivation of this work is to design an efficient constant-round scheme that overcomes communication round limitations while effectively amortizing the costs introduced by large-scale tree models in WAN setting.

TABLE I: A representative survey of private decision tree evaluation schemes for single tree.

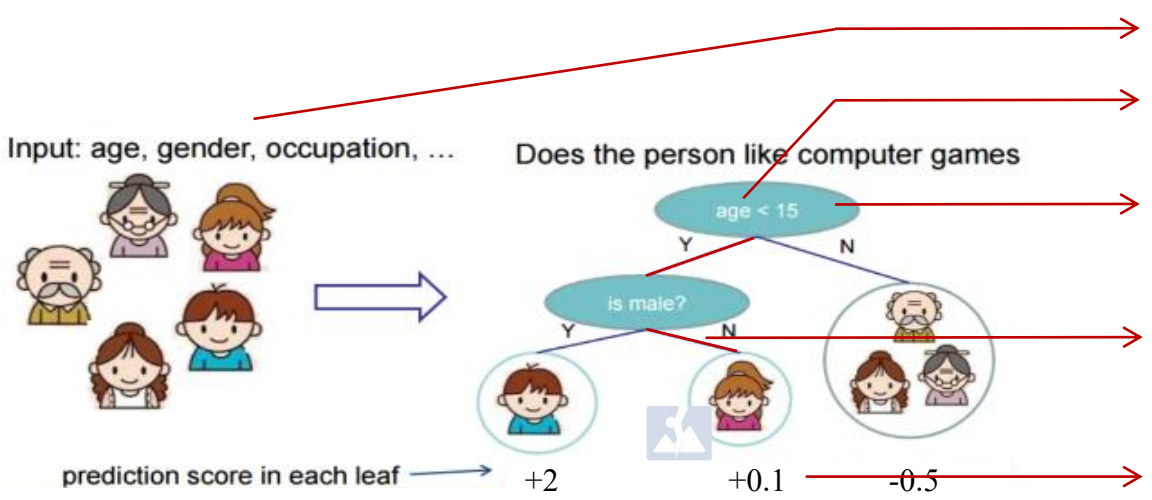
	Squirrel [43]	SGBoost [44]	FSSTree [24]	Cheng [31]	Zheng [26]	Zhao [45]	Yuan [23]	Ma [19]	Bai (HE) [20]	Tai (HHH) [14]	Kiss (HGH) [15]	Sorting-Hat [10]	Levelup [11]	Ours
Primitives	COT	LHE	RSS,FSS	RSS,FSS	SS,OT, TTP	AHE	AHE, PRF	SS,GC, OT	AHE,SS, OT	AHE	AHE,GC	FHE	FHE	PHE,SS
Round	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Feature	○	○	●	●	●	◐	●	●	●	●	●	●	●	●
Comparison	✗	$O(\tau)$	$O(D)$	$O(2^D)$	$O(2^D)$	$O(2^D)$	$O(D)$	$O(D)$	$O(D)$	$O(\tau)$	$O(\tau)$	$O(\tau)$	$O(\tau)$	$O(1)$
Path	COTPath	Poly	OnePath	OnePath*	Poly	OnePath*	OnePath	OnePath	OnePath	Path	Path	Poly	Path	MixPath
Sparse	✓	✓	✓	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓
OneTime	✓	✓	✗	✗	✓	✗	✗	✗	✓	✓	✓	✓	✓	✓
Amortized	✓	✗	✓	✓	✓	✗	✗	✓	✓	✗	✓	✗	✗	✓
Scenario	Fed	Fed	O-3PC	O-3PC	O-2PC	O-2PC	O-2PC	2PC, O-2PC	2PC	2PC	2PC	2PC	2PC	2PC, O-2PC

† Round: communication round, ●: $O(1)$; ●: $O(D)$; ●: $O(D+\log t)$; D: the maximum tree depth; t: the bit size of feature. Feature: secure feature selection, ○: no support; ◐: no oblivious; ●: oblivious. Comparison: the complexity of comparison, τ: the number of decision node. Path: the path evaluation, COTPath: get weight by COT, Poly: polynomial-based evaluation, OnePath: get weight by one path; OnePath*: get weight by one path after tree permutation; Path: Path cost-based evaluation; MixPath: Combine Path with Poly. Sparse: sparse tree model. OneTime: one-time setup phase. Amortized: calculation amortization. ✓: support; ✗: no support. Fed: inference in federated learning; O: outsourcing. In the schemes [10], [11], we only analyze the operation costs of XCMP.

- Low communication-round
- Amortizable computation
- No offline preprocessing requirement
- Sparse-model support
- Broad applicability across scenarios

Preliminaries

- **Decision Tree Evaluation**



feature vector X

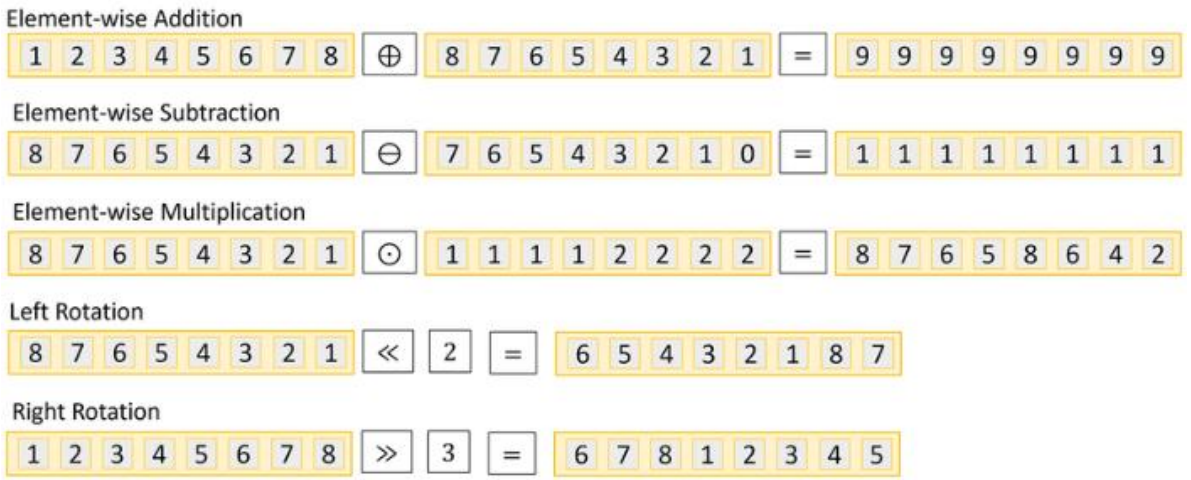
feature index m ∈ [1,M]

threshold y_{m^n}, n ∈ [1,τ]

decision path

predict result/weight w

- **Packed Homomorphic Encryption**



For large-scale operations, PHE is much more efficient than AHE, such as paillier

- N is the polynomial size
- Encryption is as $[[a]] = [[a_1],[a_2],....,[a_N]]$
- Rot([[a]], r): r > 0, right rotation

- **Private Data Comparison Algorithm**

  - The server owns two ciphertext for m_1 and m_2: $[m_1]$, $[m_2]$, How to compare $m_1 > m_2$?


Client


Server

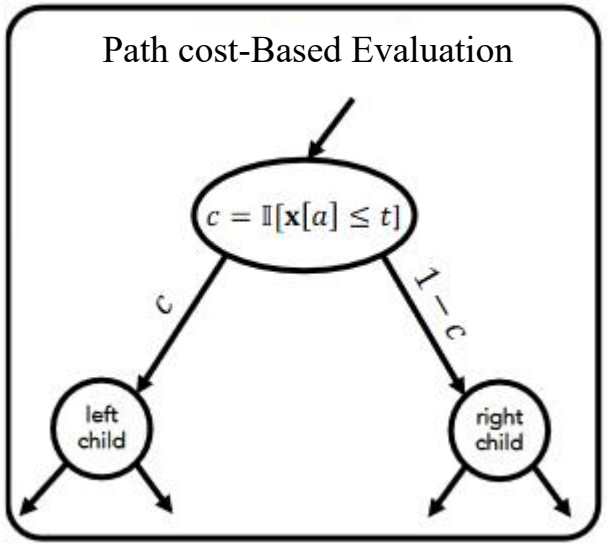
1. generate a and b, a>b>0
2. $[v] = a\ ([m_1] - [m_2])\ + b$

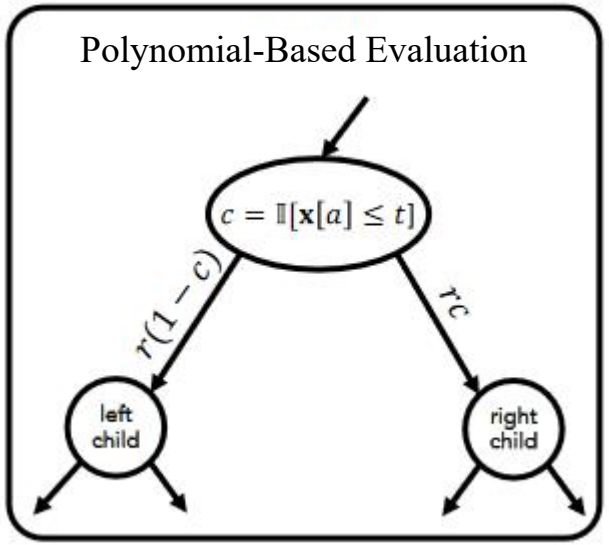
3. $v < 0\ ?\ v' = 1 : 0$

v'

The server owns the comparion result v';
The client cannot owns the comparion result m_1 and m_2.

- **Path Cost-Based and Polynomial-Based Evaluation**





Framework Overview

- **Challenges and Observations**

  Our core idea is to leverage PHE to amortize computation and communication overhead, enabling a constant-round inference scheme. Although existing approaches use PHE for model inference, they either do not fully exploit the potential of SIMD or suffer from low packing utilization and limited execution efficiency.

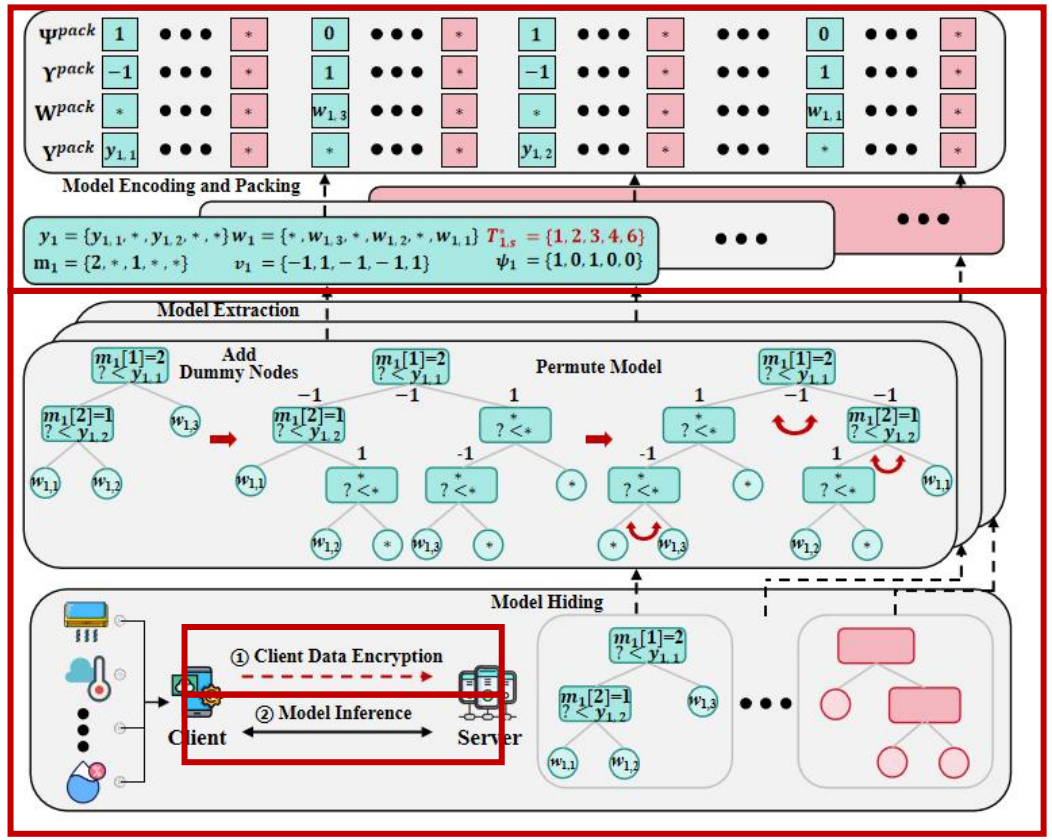
  - **Challenge (Low Packing Utilization) and Observation 1:**
    - Parallel Feature Selection and Secure Comparison by PHE
    - Each coefficient represents a decision node
  - **Challenge (Limited Execution Efficiency) and Observation 2:**
    - Path Cost-Based and Polynomial-Based Evaluation will incur huge costs
    - A novel path evaluation protocol enables path evaluation directly on plaintexts

Framework Overview

- **Kangaroo Workflow**

We illustrate the workflow of Kangaroo using a client-server model as an example, including two one-time offline operations and two online operations: Model Hiding and Extraction, Model Encoding and Packing, Client Data Encryption, and Model Inference.

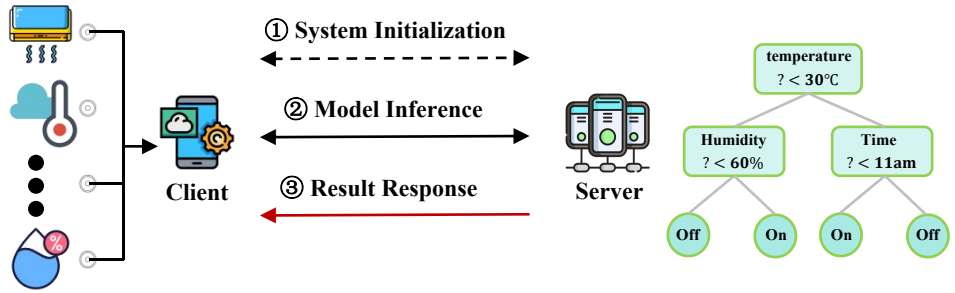
- **Model Hiding and Extraction:**
  - Obfuscate the model to protect IP and extract parameters for inference
- **Model Encoding and Packing:**
  - Quantize, pack, and encrypt model parameters once for permanent inference
- **Client Data Encryption:**
  - Encode feature vectors into polynomials and encrypt
- **Model Inference:**
  - Client and server run constant-round communication for inference



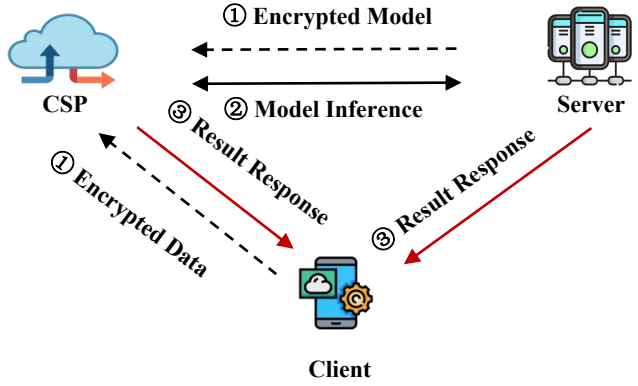
- Low communication-round
- Amortizable computation
- No offline preprocessing requirement
- Sparse-model support
- Broad applicability across scenarios

Framework Overview

- **System and Security Model**
  - The node feature and threshold
  - Model structure
  - Comparison results
  - Clients' data
  - True path
  - Evaluation result ....

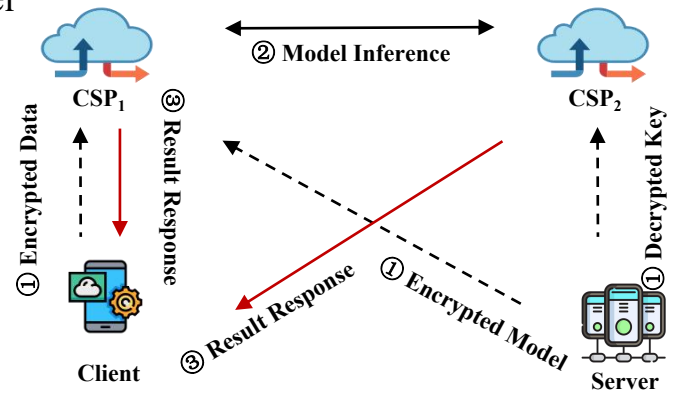


Kangaroo for Client-Server model



Kangaroo for Single-Cloud assisted model

Further extend to double-cloud model



* considering the cost of leasing servers and the security issues (Model Leakage by collusion attack) associated with the double-cloud model, we have chosen the single-cloud assisted outsourcing scheme and those interested can refer to our paper..

- **I-PackFeatureSel (No interaction)**

Algorithm 1 I-PackFeatureSel

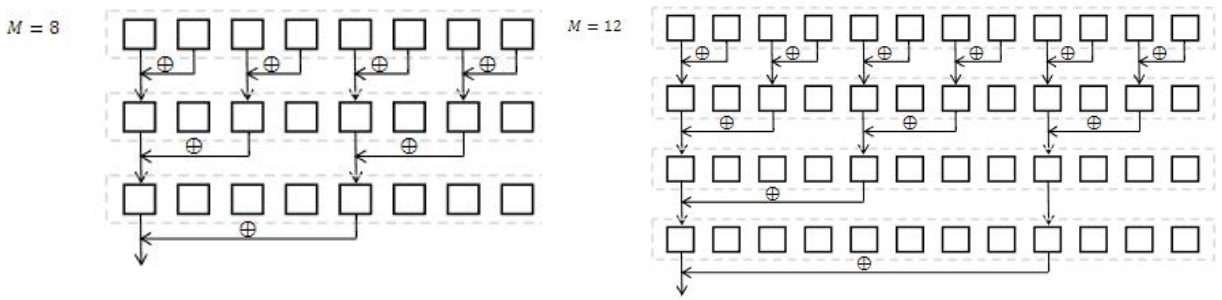
SInput: The encrypted vector $[\![X]\!]$, feature size M, and encoded feature index vector $\mathbf{M}$.

SOutput: The selected encrypted vector $[\![X']\!]$.

1: ▷ *The server executes:* ◁
2: $[\![X']\!] = [\![X]\!] \circ \mathbf{M}$.
3: $bool = (M > 0 \ \&\ (M \ \&\ (M-1)) == 0)$. ▷ *Determine whether M is a power of 2.*
4: **if** $bool == true$ **then**
5: **for** $i = 0, i < \log M, i++$ **do**
6: $[\![X']\!] = [\![X']\!] + \mathtt{Rot}([\![X']\!], -2^i)$.
7: **else**
8: $[\![X'']\!] = [\![X']\!] + \mathtt{Rot}([\![X']\!], -1)$.
9: **if** $M \mod 2 == 0$ **then**
10: $[\![X']\!] = [\![X'']\!]$.
11: **for** $i = 1, i < \lceil \log M \rceil - 1, i++$ **do**
12: $M = M - \lfloor \frac{M}{2} \rfloor$.
13: **if** $M \mod 2 == 0$ **then**
14: $[\![X']\!] = [\![X'']\!] + \mathtt{Rot}([\![X']\!], -2^i)$.
15: $[\![X'']\!] = [\![X'']\!] + \mathtt{Rot}([\![X'']\!], -2^i)$.
16: $[\![X']\!] = [\![X'']\!] + \mathtt{Rot}([\![X']\!], -2^{\lceil \log M \rceil - 1})$.
17: The server gets $[\![X']\!]$.

S	0	1	0	0	1	0	0	0	0	0	0	1
X	4	8	7	3	4	8	7	3	4	8	7	3
X' = S ∘ X	0	8	0	0	4	0	0	0	0	0	0	3
Rot(X', − 1)	8	0	0	4	0	0	0	0	0	0	3	0
X' += Rot(·)	8	8	0	4	4	0	0	0	0	0	3	3
Rot(X', − 2)	0	4	4	0	0	0	0	0	3	3	8	0
X' += Rot(·)	8	?	?	?	4	?	?	?	3	?	?	?

O(log M) Rot



- **PackObliviousCom**

Quantize:

client's data $\mathcal{X} = \{\frac{x_1 - x_1^{min}}{x_1^{max} - x_1^{min}} \cdot \zeta, \cdots, \frac{x_M - x_M^{min}}{x_M^{max} - x_M^{min}} \cdot \zeta\}$

model's feature $\mathbf{y}_k = \{\frac{y_k[1] - x_{m_k[1]}^{min}}{x_{m_k[1]}^{max} - x_{m_k[1]}^{min}} \cdot \zeta, \cdots, \frac{y_k[\tau^*] - x_{m_k[\tau^*]}^{min}}{x_{m_k[\tau^*]}^{max} - x_{m_k[\tau^*]}^{min}}$

Constraint:

$$\zeta = 2^{\frac{\log q}{2}-1}, \zeta > a > b > 0,\ 0 \le \chi,\ y_k \le \zeta$$

① The above constraints help us avoid overflow, thus ensuring correctness.

$$\chi[i] - y_k[i] \in [-\zeta, \zeta]$$

$$a[i](\chi[i] - y_k[i]) + b[i] \in (-\zeta^2 + \zeta, 0) \cup (0, \zeta^2 + \zeta)$$

$$-a[i](\chi[i] - y_k[i]) - b[i] \in (-\zeta^2 - \zeta, 0) \cup (0, \zeta^2 - \zeta)$$

Q:

$$(-\zeta^2 - \zeta, -\zeta^2 + \zeta) \to \chi[i] - y_k[i] > 0$$
$$(\zeta^2 - \zeta, \zeta^2 + \zeta) \to \chi[i] - y_k[i] > 0$$

The sign of $\chi[i] - y_k[i]$ is leaked with a certain probability.

$$\frac{4\zeta}{2\zeta^2 + 2\zeta} = \frac{2}{\zeta + 1}, \zeta = 2^{\frac{\log q}{2}-1}$$

A: ① Increase q, ② Enhancement algorithms are provided[1]. ③ Limit the number of queries ④ Randomly swap $\chi[i] - y_k[i]$

[1] https://arxiv.org/pdf/2509.03123

Client

Server

1. generate **a** and **b**, $\boldsymbol{r}$, $\zeta > a_i > b_i > 0$, $r_i = 1/-1$ by flip a coin

2. $[\boldsymbol{v}] = \boldsymbol{r}a\ ([\mathbf{m}_1] - [\mathbf{m}_2]) + \boldsymbol{r}b$

3. $v_i < 0\ ?\ v'_i = 1 : 0$

$[\boldsymbol{v}']$

$x'_i - y_i$	r_i	v_i	v'_i	c'_i	c_i
< 0	-1	> 0	1	1	0
< 0	1	< 0	0	0	0
≥ 0	-1	< 0	0	1	1
≥ 0	1	> 0	1	0	1

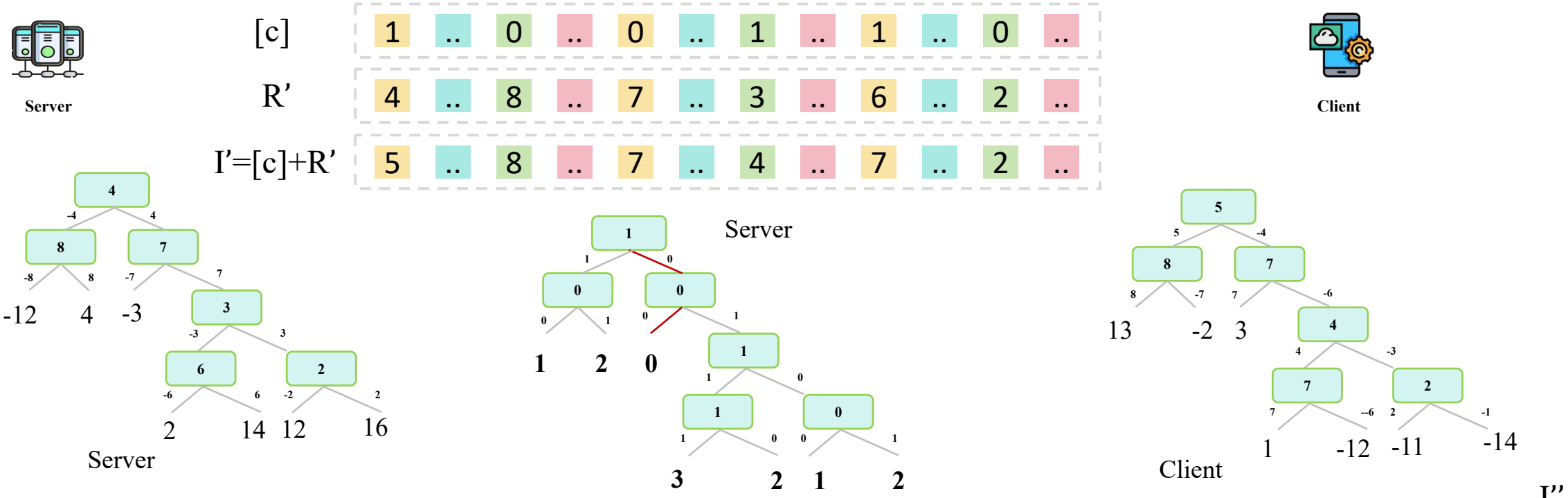
4. generate $c'_i = 1$ if $r_i = -1$, $c'_i = 0$ if $r_i = 1$

5. $[\boldsymbol{c}] = \boldsymbol{c}' + \boldsymbol{r} * [\boldsymbol{v}']$

2 plainMul + 1 Dec + 1 Enc + 1 Interaction

Packobliviouscom can finish the compasion in parallel, where the server and client cannot obtain the compaision result.

- **PackPathEva**
  - The server share the model structure with the client, and the model structure is hidden by adding dummy nodes and randomization



Building Blocks

- **PackPathEva**

Path Cost-Based


Server

[I] 1 .. 2 .. 0 .. 3 .. 2 .. 1 .. 2

r r1 .. r2 .. r3 .. r4 .. r5 .. r6 .. r7

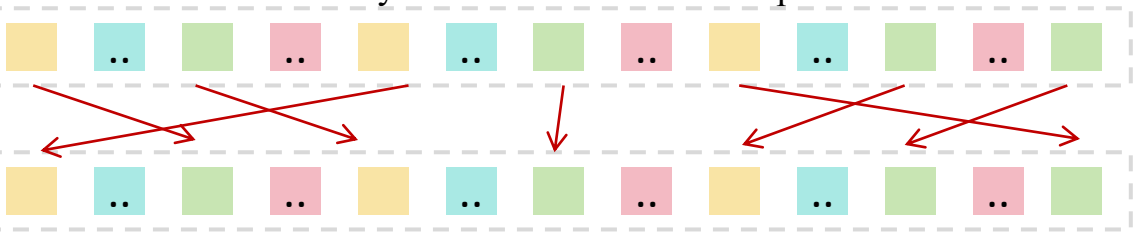
[I] 1 .. 2 .. 0 .. 3 .. 2 .. 1 .. 2

k k1 .. k2 .. k3 .. k4 .. k5 .. k6 .. k7

W w1 .. w2 .. w3 .. w4 .. w5 .. w6 .. w7

The shuffle function is ulitilized to hide the true weight, which is achieved by lots of rotation and multiplication.

Shuffle



Select the position with 0.
The corresponding weight is selected

shuffle([I]*r)

shuffle([I]*k+w)


Client

Algorithm 4 PackPathEva

Input: The encrypted vector $[\![C]\!]$, the structure index $\mathcal{T}_s$ and the encoded weight $\mathbf{W}$.
Output: The encrypted evaluation result vector $[\![T]\!]$.

1: ▷ *The server executes:* ◁
2: $R' \leftarrow \{r'_1, *, \cdots, *, \cdots, r'_{\tau^*}, *, \cdots, *\}$, $[\![I']\!] \leftarrow [\![C]\!] + \mathbf{R}' \Rightarrow$ the client.
3: ▷ *The client constructs a tree by $\mathcal{T}_s$ and executes:* ◁
4: $I' \leftarrow \mathtt{Dec}([\![I']\!], \mathtt{s})$, the $i'_n \leftarrow n$-th `node->left.cost` and the $1 - i'_n \leftarrow n$-th `node->right.cost` for $1 \le n \le \tau^*$.
5: $I'' \leftarrow \{i''_1, 0, \cdots, 0, \cdots, i''_{\tau^*+1}, 0 \cdots, 0\}$, where i''_n is the sum of cost along the n-th path and $1 \le n \le \tau^* + 1$.
6: $[\![I'']\!] \leftarrow \mathtt{Enc}(I'', \mathtt{pk}) \Rightarrow$ the server.
7: ▷ *The server constructs a tree by $\mathcal{T}_s$ and executes:* ◁
8: The $-r'_n \leftarrow n$-th `node->left.cost` and the $r'_n \leftarrow n$-th `node->right.cost` for $1 \le n \le \tau^*$.
9: $R'' \leftarrow \{r''_1, 0, \cdots, 0, \cdots, r''_{\tau^*+1}, 0 \cdots, 0\}$, where r''_n is the sum of cost along the n-th path and $1 \le n \le \tau^* + 1$.
10: the server gets the encrypted results of path cost-based evaluation $[\![I]\!] \leftarrow [\![I'']\!] + \mathbf{R}''$.
11: ▷ *The server and client jointly execute:* ◁
12: The server gets $[\![C']\!] \leftarrow \mathtt{PackObliviousCom}(-[\![I]\!]^2)$. ▷ *Transform path cost-based into polynomial-based.*
13: ▷ *The server executes:* ◁
14: The server gets the evaluation result $[\![T]\!] \leftarrow [\![C']\!] \circ \mathbf{W}$.

Our approach

Take -I as the input of PackObliviousCom


Server

[C'] 0 .. 0 .. 1 .. 0 .. 0 .. 0 .. 0

W w1 .. w2 .. w3 .. w4 .. w5 .. w6 .. w7

Our Optimizations

- **The same sharing for same model**

The server runs PackHiddenMod to obtain $\mathcal{T}_s^*$ and $\{\mathbf{Y}, \mathbf{S}, \mathbf{W}, \Upsilon, n^*\}$.

Our Observation:

The client cannot distinguish which leaf node is accessed from the paths

→ The model can be reused.

↓

N, q is fixed

↓

The encoder is fixed → $\{\mathbf{Y}, \mathbf{S}, \mathbf{W}, \Upsilon, n^*\}$ can be reused

the corresponding keys and $\mathcal{T}_s^*$ ←

encrypted model $\{[\![Y]\!], [\![S]\!], [\![W]\!], [\![\Gamma]\!]\}$

The decrypt key is owned by server

↓

The encrypted model can be reused.

- **Latency-Aware Strategy**

1. The generation of these parameters, such as $\{\mathbf{E}, \mathbf{E}', \mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{R}', \mathbf{R}''\}$ is independent of the model or the client's data.
2. The client (PME) or the server (OPME) needs to excute the encryption operations.
3. Some operations can be merged.
4. Minimize the ciphertext operation.

[[0]]

$\{\mathbf{E}, \mathbf{E}', \mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{R}', \mathbf{R}''\}$

$B \circ R - A \circ R \circ Y$

$-A \circ R$

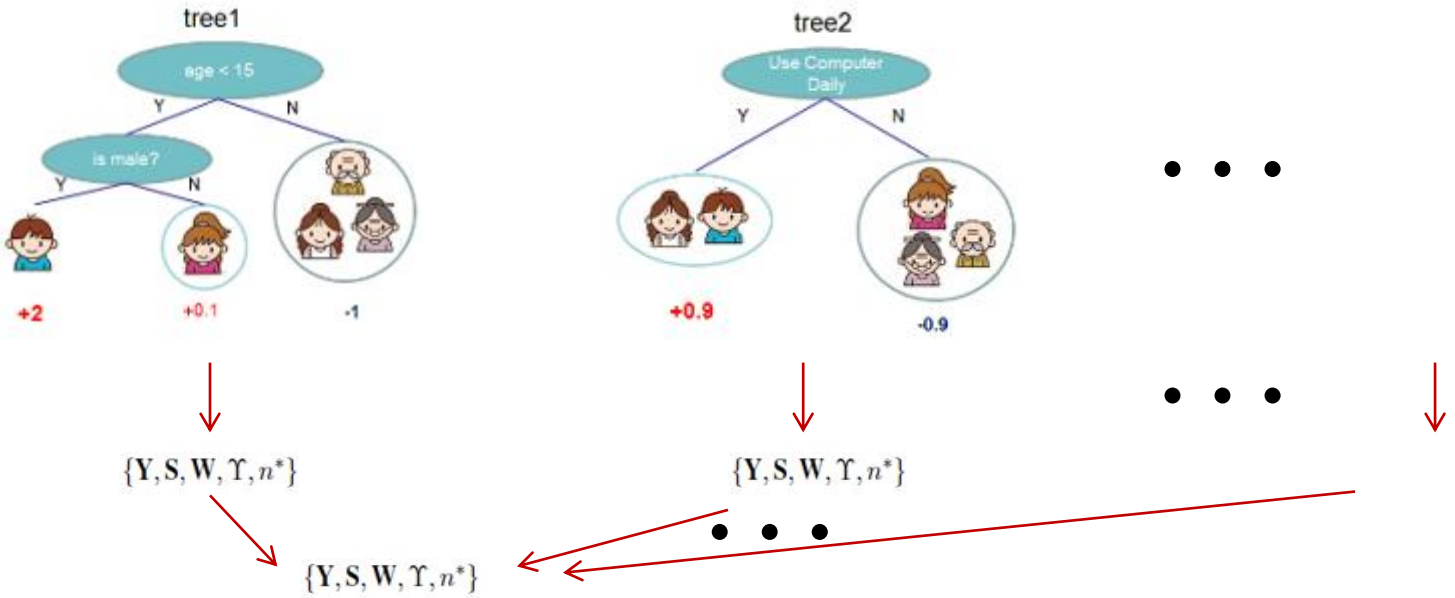
[[0]]

$\{\mathbf{E}, \mathbf{E}', \mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{R}', \mathbf{R}''\}$

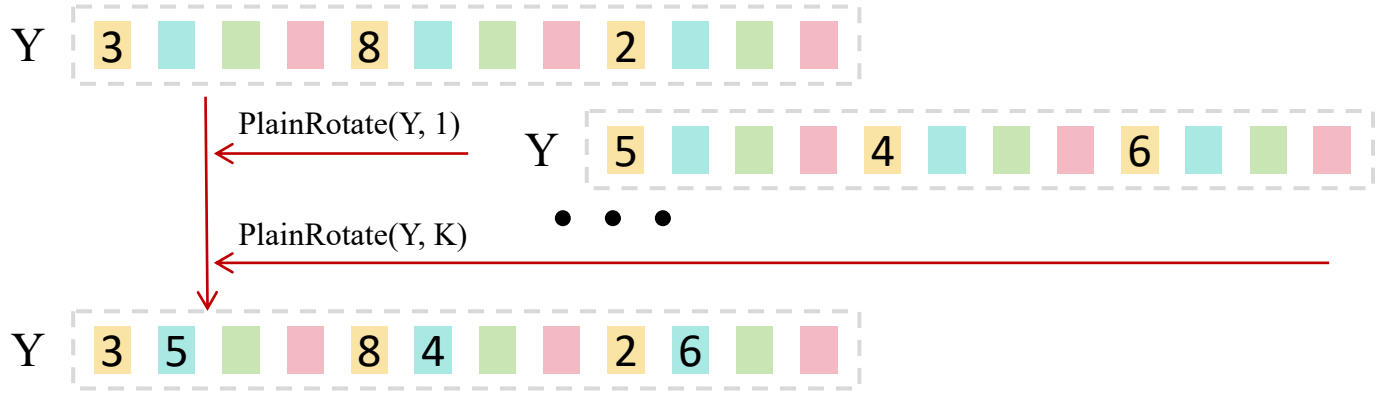
$-A \circ R$

Our Optimizations

- **Adaptive Encoding Adjustment**



Example for Y



- **Adaptive Encoding Adjustment**

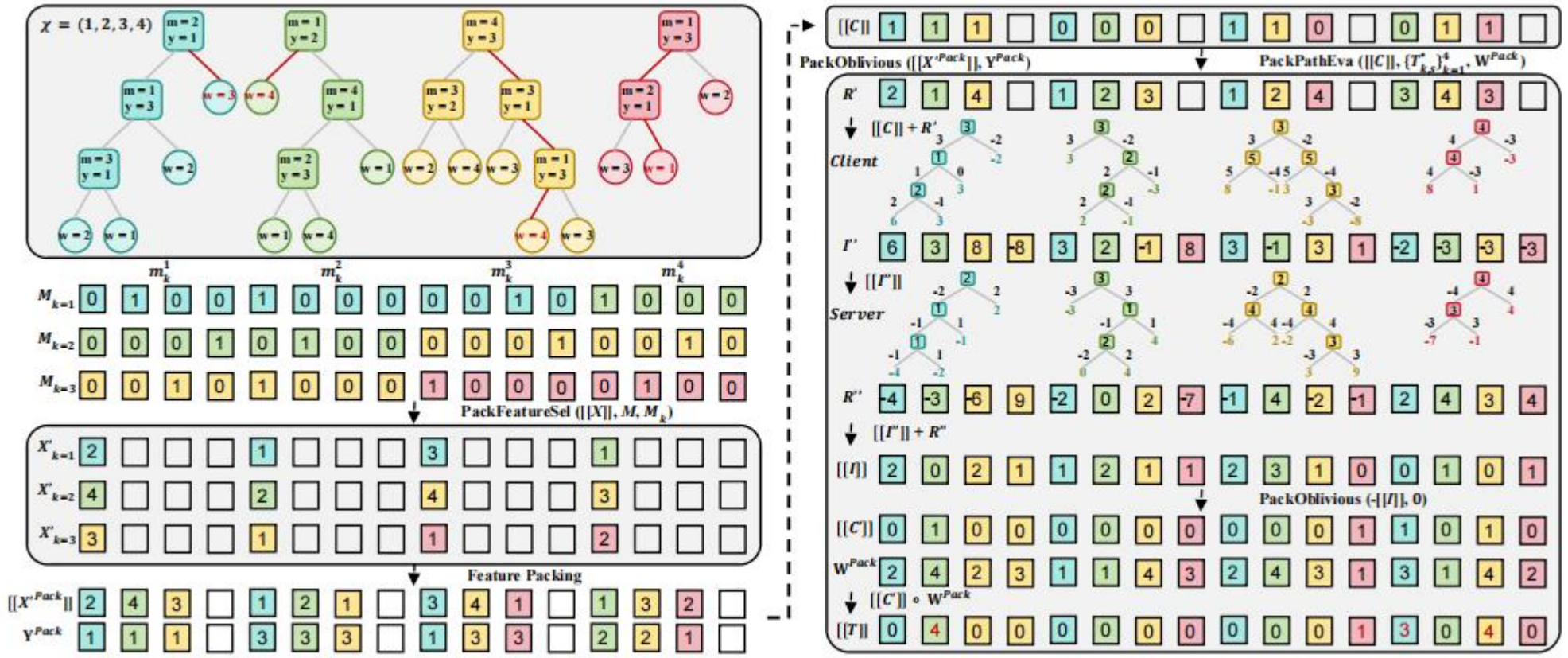


Fig. 3: Toy example without model hiding for adaptive encoding adjustment, where $\mathtt{N} = 16$, $M = 4$, and $K = 4$.

Our Optimizations

- **Complexity Analysis**

TABLE II: Online operation costs for two-party inference schemes: τ : the number of decision nodes; M : the feature dimension; t : the bit size of feature; K : the number of trees; D : the tree depth; $\texttt{Mul}^*$: plaintext-ciphertext multiplication. SOS: Shard Oblivious Selection Protocol [20]; DGK: Private Comparison Protocol [83].

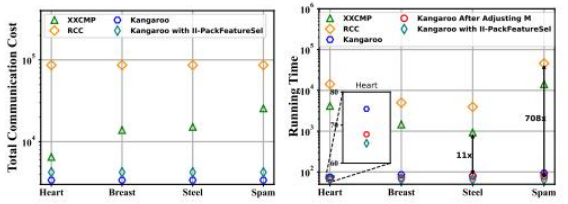
	Primitives	**Selection**	**Comparison**	**Path evaluation**	**Round**
Ma [19] (Sparse)	SS,GC,OT	$KD\cdot$(SS, $\binom{M}{1}$-OT)	$KD\cdot$ (GC, $\binom{2}{1}$-OT)	/	2D-1
Bai [20] (HE-SOS)	SS,OT,AHE	$KD\cdot$ SOS	$KD\cdot$ (GC, SOS)	/	8D
Cong [10] (Sortinghat)	PHE	/	$K\tau\cdot\texttt{Mul}^*$	$O(K\tau)(\texttt{Mul}+\texttt{Add})$	1
Mahdavi [11] (Levelup)	PHE	/	$K\tau\cdot\texttt{Mul}^*$	$O(K\tau)(\texttt{Add}+\texttt{Mul}^*)$	1
Tai [14] (HHH)	AHE	/	$K\tau\cdot$ DGK	$O(K\tau)(\texttt{Add}+\texttt{Mul}^*)$	2
Kiss [15] (GGH)	GC,AHE,OT	$KMt\cdot$(GC + $\binom{2}{1}$-OT)	$K\tau\cdot$ GC	$O(K\tau)(\texttt{Add}+\texttt{Mul}^*)$	2
Kiss [15] (HGH)	GC,SS,AHE	$KM\tau\cdot(\texttt{Add}+\texttt{Dec})$	$K\tau\cdot$ GC	$O(K\tau)(\texttt{Add}+\texttt{Mul}^*)$	3
Ours (Kangaroo)	**PHE,SS**	$2K\cdot\texttt{Mul}^*+O(K\log M)\texttt{Rot}$	$\lceil\frac{K}{M}\rceil(2\cdot\texttt{Mul}^*+\texttt{Enc}+\texttt{Dec})$	$2\lceil\frac{K}{M}\rceil(\texttt{Mul}^*+\texttt{Enc}+\texttt{Dec})$	**4**

[1] In the schemes [10], [11], we only analyze the operation costs of XCMP.

Performance Evaluation

- **Testing Environment**
  - Real-world WAN conditions:
    - Client: i5-9400H 2.50GHz CPU, 23.1GB RAM, Ubuntu 18.04.6
    - Server: JD cloud and Alibaba cloud
  - Simulated network conditions:
    - LAN, 1 Gbps, RTT 0.1 ms
    - MAN, 100 Mbps, RTT 6 ms
    - WAN, 40 Mbps, RTT 80 ms
  - Our code can be see in

    https://github.com/pigeon-xw/Kangaroo
- **Results**
  - 14–59× faster than state-of-the-art one-round schemes on small datasets
  - Less impacted by tree structure compared to multi-round and deep-related schemes
  - Seconds-level latency in WAN for large-scale tree evluation



(a) Total Communication Cost (b) LAN (1Gbps/0.1ms)

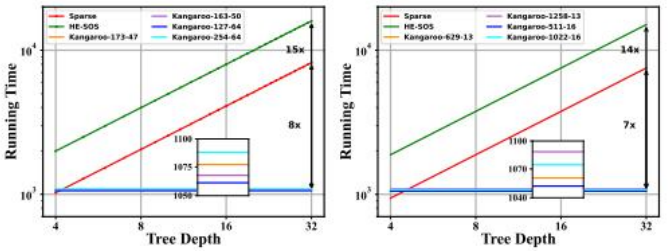
(c) MAN (100Mbps/6ms) (d) WAN (40Mbps/80ms)

Fig. 5: The comparison of total communication cost (KB) and running time (ms) with one round interactive schemes. The (M, D, T) are as follows: Heart $(13, 3, 5)$, Breast $(30, 7, 17)$, Steel $(33, 5, 6)$, and Spam $(57, 16, 58)$.

TABLE VII: Total online runtime for privacy-preserving applications under real-world WAN conditions.

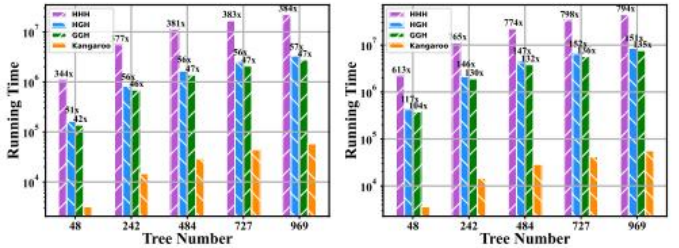
Applications	JD Cloud (5 Mbps), TP (1 Mbps), RTT (30 ms)	Ali Cloud (100 Mbps), TP (1 Mbps), RTT (40 ms)
Image Recognition	6.96 (s)	3.18 (s)
Medical Diagnostics	5.92 (s)	2.52 (s)
Financial Forecasting	5.98 (s)	3.06 (s)

[1] We leverage the Digits $(47, 15, 168)$, Diabetes $(10, 28, 393)$, and Boston datasets $(13, 30, 425)$ to demonstrate Kangaroo's applicability in image recognition, medical diagnostics, and financial forecasting, respectively.



(a) $M = 47$ (b) $M = 13$

Fig. 6: The running time (ms) for large-scale single tree evaluation across different tree depths in WAN setting. Kangaroo-a-b: where a denotes the maximum number of decision nodes supported, and b represents the adjusted feature dimension M.



(a) Digits (b) Boston

Fig. 7: The running time (ms) for large-scale random forest evaluation across different tree numbers in WAN setting.

Conclusion

- **Our contribution:**
  - We propose a novel client–server architecture for decision tree inference that is minimally affected by changes in model structure and is well suited for evaluation in large-scale settings over WAN.
  - A novel set of secure components is designed to support efficient decision tree evaluation.
  - A modular design is provided[1] to allow for modifications and also provides suggestions for different scenarios.
    - ✓ I-PackFeatureSel → OT for high-dimensional data
    - ✓ PackObliviousCom → Polynomial approximation, Path evaluation → Path Cost-Based for non-interactive + GPU
    - ✓ Kangaroo → NDSS 21[2] for small-scale decision trees over LAN

Email: xuwei_1@stu.xidian.edu.cn

[1] https://github.com/pigeon-xw/Kangaroo
[2] Ma J P K, Tai R K H, Zhao Y, et al. Let's stride blindfolded in a forest: Sublinear multi-client decision trees evaluation[J]. 2021.

Thank you