Enhanced Outsourced and Secure Inference for Tall Sparse Decision Trees

Andrew Quijano, Amazon[a] & New York University, U.S.A.
Spyros T. Halkidis, University of Macedonia, Greece
Kevin Gallagher, NOVA LINCS &
NOVA School of Science and Technology, Portugal
Kemal Akkaya, Florida International University, U.S.A.
Nikolaos Samaras, University of Macedonia, Greece

[a]The work was done outside of his role at Amazon.

Decision Trees

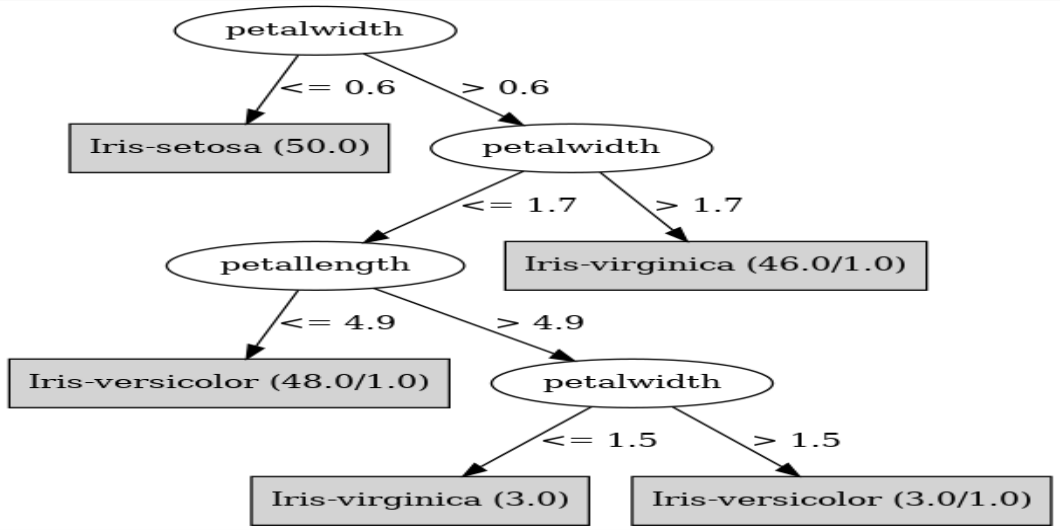
- Decision Tree (DT) an easy to understand tool that has been widely used for classification tasks.
- A DT consists of internal nodes with a threshold value that determines which node to go to next.
  - All classes are on the leaf nodes of the DT.
- DTs are useful as you can audit why a classification was made
- DTs are used managing student data, health data, and fraud detection

Sparse Decision Trees

- In a sparse DT, most of the leaves in it are not at the maximum depth of the tree
- To analyze if a DT is sparse, we can analyze the level of the output classifications from the training dataset

Dataset	Average	Median	3rd Quartile	Max	Size
Hypothyroid	2.78	2	2	9	3372
Spambase	9.60	9	11	22	4601
Nursery	5.99	7	8	14	12960

Spare Decision Tree Example - Iris



Secure Inference

- A Privacy-Preserving Decision (PPDT) should satisfy the following:
  - We want to be able to have a client provide an input to evaluate the DT without revealing the input
  - The DT owner would want to have clients evaluate it but not have their model reverse engineered by the client
- PPDTs also enables compliance with privacy regulations such as
  - Health Insurance Portability and Accountability Act (HIPAA) in the U.S.
  - General Data Protection Regulation (GDPR) in the E.U.

Preliminaries - Homomorphic Encryption (HE)

- Additively Homomorphic Encryption: that uses an Encryption algorithm **Enc** is one such that for any two messages α, $\alpha^{'}$ in the plaintext space $\mathcal{M}$ it holds that **Enc**$(\alpha + \alpha^{'})$=**Enc**$(\alpha) \boxplus$ **Enc**$(\alpha^{'})$ (we use the 'boxplus' operator ($\boxplus$) to denote the addition of two ciphertexts). For simplicity, as it is common in many papers, we write an encryption of α as $[\![\alpha]\!]$ instead of **Enc**(α).
- Socialist's Millionaire's problem (secure integer comparison protocol): First stated by Yao in 1982. The issue is about two millionaires wanting to compare their wealth **without** revealing their net worth
- To securely compare two encrypted integers, assume that there are two integers held by A and B. Suppose that A has the t-bit integer $x = \sum_{i=0}^{t-1} x_i 2^i$ in binary form, while party B has another t-bit integer $y = \sum_{i=0}^{t-1} y_i 2^i$. The goal is for A and B, respectively, to obtain the protocol bits δ_A and δ_B, such that $\delta_A \oplus \delta_B = \mathbb{1}\{x \geq y\}$.

Problem Motivation

- Prior research used techniques such as homomorphic encryption to create PPDTs
- One approach is by Joye and Salehi, in which they implement a new timing attack resistant encrypted integer comparison protocol, and they convert the DT to a **complete** binary tree so the PPDT has strong timing attack resistance since there will always be the same number of encrypted comparisons.
- **Issues:** Joye and Salehi's approach performs poorly in classification time for sparse DTs. Also, it has no good redundancy as the model is stored on one server.

Related Work - Liu et al.

- Utilize the Distributed Two Trapdoors Public Key Cryptosystem (DT-PKC), allowing comparisons between ciphertexts encrypted using different public key pairs.
- Liu et al. created new protocols such as securely count the frequency of attributes in the dataset, used for training the DT on encrypted data.
- Our PPDT uses the Joye and Salehi's timing attack resistant comparison protocol, and has faster evaluations than Liu et al. despite using a larger key size.

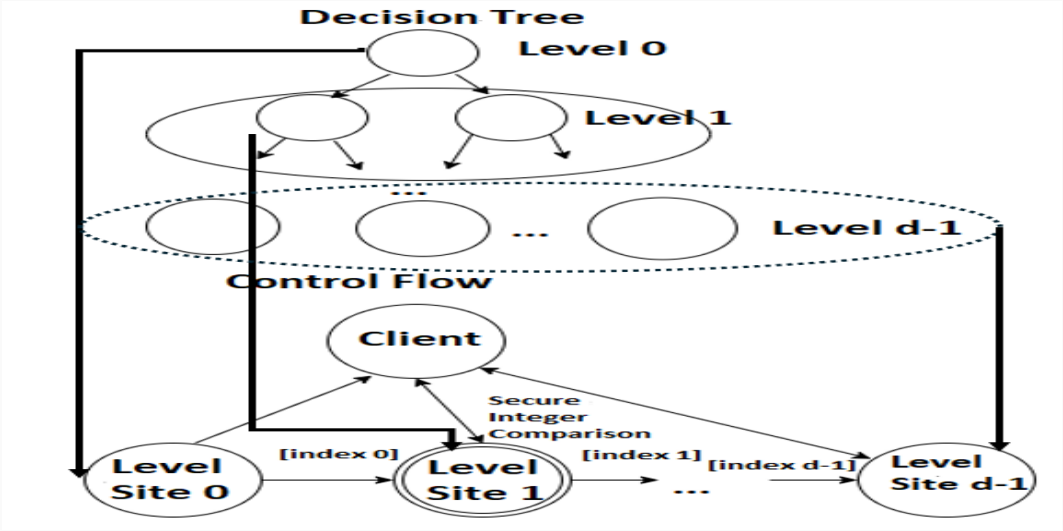
Related Work - Yuan et al.

- Yuan et al. utilize gradient-boosting decision trees (GBDT) to train DTs
- They introduce the privacy preserving distributed GBDT (PPD-GBDT), which uses differential privacy, polynomial approximation and fully homomorphic encryption.
- For evaluation, the client encrypts their input and sends the ciphertext and the public key. The client obtains and decrypts the classification result.
- Our PPDT has generally faster PPDT evaluation times as we already encrypt everything before evaluation. Also, no noise is added, which could potentially cause classification inaccuracies.

Our Approach

- We assume the DT exists on a server, and wants to outsource the DT model to the cloud.
- We split the DT model into level-sites, which has all the nodes for each specific level
- The level-sites will use a comparison protocol and track which nodes are in-scope for the next level-site
- The client will run the comparison protocol with each level-site, until a leaf is reached and the level-site returns the classification

Our Approach - Visualized



Threat Model

- Threat Model: We assume that the server is trusted but the client and level-sites are "honest-but-curious" (HBC), specifically:
  - We assume the client will attempt to learn the thresholds of the PPDT from the level-sites
  - We assume the level-sites will attempt to learn both the input vector and classification
  - We also consider a passive adversary attempting to learn the client's input, classification and PPDT thresholds
- We also assume that the level-sites will attempt to use power-based side channel attacks and timing attacks

PPDT Initialization

- Client creates key pair and gives public key to the server
- Server trains DT model and encrypts thresholds and leaves
- Server distributes encrypted data to the level-sites
- Server provides the client with a label encoder for encrypted equality checking for comparing categorical data

Level-Site Evaluation

- Let T_k^l be the threshold at level l, index k is the index of the node at level l. x_i is the appropriate feature value to compare with.
- A comparison protocol is run between the client and level-site. The level-site provides $[\![T_k^l]\!]$ and $[\![x_i]\!]$ and gets the boolean value of an inequality.
- Use Joye and Salehi comparison protocol, $\mathbb{1}\{T_k^l \leq x_i\}$, for numerical data, and a modified, timing attack resistant Veugen encrypted equality comparison protocol, $\mathbb{1}\{T_k^l == x_i\}$, for categorical data.

Security Analysis

Security analysis is performed by assuming HBC.

- Security against an HBC client.
  - Client throttling
    - The level-sites can limit number of client requests to prevent thresholds being reverse engineered
    - The level-site returning classification can add extra delay so the client won't know if the classification was at level-site l or $l + 1$
  - Input vector is secure due to HE
- Security against an HBC level-site l.
  - Level-site has access only to encrypted data
  - Index has no relation to encrypted data
- Security Against Side Channel Attacks.
  - Can split level-sites between two non-colluding providers
  - Containers offer strong power-based attack mitigations and make key rotation easier
  - All level-sites only complete 1 encrypted comparison, and they use a timing attack resistant comparison protocol

Experiment Setup

- We implemented and tested the proposed approach using Amazon Elastic Kubernetes Service (EKS). We used t2.medium EC2 instances, which have 2 vCPUs and 4 GB RAM.
- Our open source code implementation used a homomorphic encryption library implemented in Java that implements Joye and Salehi's encrypted integer protocol and Veugen's encrypted equality comparison.
- We used Docker containers and Kubernetes, as it allows the creation of replicas and supports horizontal scaling.

Experimental Results compared with Joye and Salehi

- The average depth for the Hypothyroid set is 2.78 which corresponds to a value between 0.807 and 1.772 seconds. This is significantly faster than the Joye and Salehi approach.
- Similarly, the average depth for Nursery data is 5.99 which corresponds to a value between 2.041 and 2.950 seconds, again much less than 4.140 of Joye and Salehi.

Levels	Hypothyroid Our approach	Hypothyroid Joye & Salehi	Nursery Our approach	Nursery Joye & Salehi
2	**0.807 s**	3.279 s	**0.656 s**	4.140 s
4	**1.772 s**	3.279 s	**2.041 s**	4.140 s
9	4.148 s	3.279 s	**2.950 s**	4.140 s
12	N/A	N/A	5.648 s	4.140 s

Experimental Results compared with Yuan et al.

- Our approach is faster at almost all levels due to Yuan et al.'s need to encrypt data for each evaluation and the computational overhead of traversing the PPD-GBDT.

Levels	Our Approach	Yuan et al.
2	1.927 s	0.49 s
3	2.736 s	7.58 s
4	3.937 s	8.35 s
5	4.638 s	17.33 s
6	5.783 s	19.37 s

Conclusion

- We present a new secure decision tree inference protocol by splitting the DT model into level-site constructs. Our PPDT exhibits optimal performance in sparse trees due to fewer encrypted integer comparisons, resulting in faster performance compared to other PPDTs.
- It offers resistance to timing attacks through timing attack resistant comparison protocols.
- Our PPDT also offers strong mitigations against power-based side channel attacks, as we use containers that offer power-based side channel protections, key replacement.
- Our approach can be used to split our PPDT with two non-colluding cloud providers.
- Finally, our PPDT protocols allow us to support load balancing to more evenly distribute more computing resources to higher levels of the PPDT, which would experience more usage and redundancy in case a level-site experiences an outage.

Acknowledgements

- We would like to thank Professor Dimitrios Hristu-Varsakelis for helping at the initial stages of the paper by making suggestions.
- This work is supported by NOVA LINCS ref. UIDB/04516/2020 (https://doi.org/10.54499/UIDB/04516/2020) and ref. UIDP/04516/2020 (https://doi.org/10.54499/UIDP/04516/2020) with the financial support of FCT.IP.