
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 4, pp. 1065–1076, DOI 10.12694/scpe.v24i4.2220

CONFIDENTIAL TRAINING AND INFERENCE USING SECURE MULTI-PARTY COMPUTATION ON VERTICALLY PARTITIONED DATASET

KAPIL TIWARI* NIRMALYA SARKAR † AND JOSSY P GEORGE‡

Abstract. Digitalization across all spheres of life has given rise to issues like data ownership and privacy. Privacy-Preserving Machine Learning (PPML), an active area of research, aims to preserve privacy for machine learning (ML) stakeholders like data owners, ML model owners, and inference users. The Paper, CoTraIn-VPD, proposes private ML inference and training of models for vertically partitioned datasets with Secure Multi-Party Computation (SPMC) and Differential Privacy (DP) techniques. The proposed approach addresses complications linked with the privacy of various ML stakeholders dealing with vertically portioned datasets. This technique is implemented in Python using open-source libraries such as SyMPC (SMPC functions), PyDP (DP aggregations), and CrypTen (secure and private training). The paper uses information privacy measures, including mutual information and KL-Divergence, across different privacy budgets to empirically demonstrate privacy preservation with high ML accuracy and minimal performance cost.

Key words: Privacy-Preserving Machine Learning (PPML), Vertically Partitioned Datasets Secure Multi-Party Computation (SMPC), Confidential Inference, Differential Privacy (DP)

1. Introduction. Today's evolved world is continuously reinventing itself. As globalization and digitalization with Machine Learning and Artificial Intelligence have brought countries closer, personal space and privacy boundaries have been blurred. Digitalization has led to the rat race of countries trying to computerize their citizens' data. This digitalization has unlocked several productivity, accuracy, and efficiency avenues while raising the dreaded privacy question.

While Oxford defines privacy as "the state of being alone and not watched or disturbed by other people", privacy has evolved over the decades; concerning machine learning, privacy is not only of the data owners but also of model owners and clients[23]. A data owner is an individual or an organization willing to share personal data to facilitate the creation of Machine Learning (ML) applications. A model owner is a collective term used to describe the inventors of the ML model. Clients are the end-user and consumers of the Model. True privacy is one where the privacy of all these contributors is maintained and safeguarded[23].

Several methods and protocols have been developed to achieve privacy for all the contributors, and collectively these mechanisms are called Privacy-Preserving Machine Learning (PPML)[20, 18, 16, 14]. The complexity of PPML only increases when various forms of data are accessed in multiple formations. Under conventional thoughts, data is collected from various sources and owners and then combined horizontally as incremental records. Here different sources give us more data points to test, but this is not the only way. Most of the available work in the research fraternity is around Regression problems and Neural networks[7, 3, 19]. Classification, especially Gradient Boosting[2, 12, 18], has recently gained immense popularity. All the methodologies mentioned earlier are being deployed in recommendation systems, anomaly detection, predictive analytics and many more. They all simultaneously face a considerable challenge: tackling Vertically Partitioned Datasets.

To begin with, Vertical Partitioned Data can be distinguished from Horizontal Partitioned Data based on observation and features. Observations are distributed amongst various data owners/providers in Horizontal Partitioned Data for the same features. Contrary to this, in vertically partitioned data, various entities own different features/attributes of information for the same set of entities[10].

Often, individual data held by the institution cannot bring about drastic changes, but combined with other sources, it can give deeper insights. In most countries, there exists a digital database of the citizen. In the

*Department of Computer Science, CHRIST University, India (kapiltiwarires1@outlook.com)
†Department of Computer Science, CHRIST University, India
‡Department of Computer Science, CHRIST University, India

case of India, the Adhaar id acts as a unique identifier for such a database. Policy planners can get more significant insights from it when analyzed alongside the health and insurance and banking sectors' databases. These insights can help create impactful policies and social welfare; one can develop policy recommendation systems if done right. In this example, the Adhaar database, healthcare database and finance database are various data sources catering to the same set of users but having different data. This is a classic example of vertically partitioned data. Even though insights from vertically partitioned data can be of great value, it also poses significant threats to the privacy and performance of ML models.

Policy making is not the only domain wherein vertically partitioned data is utilized; several other fields like fintech and security are interested. Their primary concern is preserving all privacy: the clients that will eventually use the services, the model owner or the creator of ML applications, and the data owner.

Attempts to maintain privacy at various levels have resulted in multiple methodologies and practices. In the aggregation stage, privacy is attempted to be safeguarded via anonymization, Homomorphic Encryption, and Differential Privacy[1]. In the training and inference stages, homomorphic Encryption, Differential Privacy are again popular alongside Secure Multiparty computation techniques. Federated Learning[23] is another commonly used method that has gained popularity, especially regarding vertical partitioning datasets.

Anonymization removes information likes Name, Adhaar number, and address. This method is not functional with Vertically partitioned datasets as PII becomes key in aligning the datasets from various sources. Homomorphic Encryption (HE)[24] is the process of having encrypted data. Even though HE is effective, it comes with high computational costs and time complexities. Federated Learning is training a centralized ML algorithm on decentralized data. Here, the model is shared across various clients and is trained locally. Federated Learning has several challenges, including that expensive communication and system heterogeneity. As the model is being shared with multiple data owners, continuous communication is necessary, but transmission via a network is slower than local computation. Moreover, there exists system heterogeneity, which poses a threat. Data owners vary in hardware, connectivity and power. This can lead to unwanted connection breaks and privacy threats[25].

Based on the statistical method, Differential Privacy (DP) adds perturbation to increase privacy. Secure-Multi-Party Computation (SMPC)[21], uses collaborative computing technology with multiple parties to solve privacy concerns.

Incomplete feature information retained by a single participant and a challenging training procedure are significant issues with vertically partitioned data and needs quick resolution[9]

This paper presents the solution for preserving privacy for collaborative machine learning stakeholders using secure multi-party computation techniques, especially for the vertically partitioned dataset. The method, named CoTraIn-VPD and implemented using the Python language and a few open-source libraries, has showcased the effectiveness in preserving the privacy of collaborative machine learning stakeholders, including data owners, model builders, and inference clients. The paper describes the specific validation mechanisms to prove the privacy gain with CoTraIn-VPD using information metrics such as mutual information and KL-Divergence.

The paper is organized as follows. The context of the study is discussed in the next section, emphasizing previous research and any gaps. While section 3 describes the technique using motivation, architecture, execution flow, implementation details, and lab setup. Section 4 covers the findings of the experiments and discusses the insights and results in fact, followed by a conclusion.

2. Background. Vertically partitioned dataset consists of multiple data owners having mutually exclusive columns, features, or variables for a given population. In a real-life scenario, data is split across multiple data providers such as local and government agencies; Unique Identification systems like Aadhar, Income Tax records, life insurance firms, and transport records carry different information about an individual, which can be collaborated to produce a qualitative machine learning model to generate meaningful insights. There are multiple secure and private ways this collaboration over the vertically partitioned dataset can be carried out, like certain data aggregation techniques like anonymization; however, these techniques are not fool-proof and can still leak private information. Differential Privacy techniques can increase the privacy measured by a privacy budget but adversely affect training and inference performance apart from impacting model accuracy[23]. Some of the proposals presented in the past to solve the problem needed more empirical findings and implementations. Hell at el.[11] Came up with the first implementation using linear regression over Secure Multi-Party Computation

but used the Homomorphic Encryption technique that was too expensive to be implemented practically due to heavy computations. There was another implementation by Bogdanov et al. [4], But it was limited to 10 features. Charlotte et al.'s [5] algorithm to train a logistic regression model on an encrypted dataset using a homomorphic encryption technique proved inefficient in model accuracy and training time.

There were a few HE-based techniques for secure training and inference, like Chen et al., Li and Sun[13], Carpov et al.[6], [15], but all of them either lacked model accuracy or performance or were too expensive to implement. To securely train the partitions, there were few federated learning-based techniques like Hardy et al. [17] They applied additive HE to train the model but could not improve the loss of model accuracy due to approximation techniques. Mandal et al. [17] built a regression model using an additive secret-sharing process over high dimensional data but was limited to the horizontally distributed dataset. Liu et al. [8] established a platform to support different systems in developing ML models collaboratively over vertically partitioned datasets. Cock et al. [8] suggested a securely trained LR model for dispersed parties but used a trusted third-party initializer to assign random weights across two computing servers.

PPML consists of private training and inference and should be dealt with separately using secure multi-party computation techniques to preserve compute cost and keep the accuracy loss in control. Today, implementation has yet to empirically prove the effectiveness of a fast multi-party computation technique, especially for the vertically partitioned dataset setting.

To summarize, preserving privacy during model training and inference for machine learning stakeholders, like owners of data and models and inference clients, is an active area of research where existing work lacks implementation and empirical findings.

3. Motivation and Process.

3.1. Motivation. Vertically partitioned data spread across multiple data owners need an efficient and accurate privacy-preserving technique for model training and inference. The solution should scale with the number of partitions and model owners and cater to multiple inference users quickly, securely, and privately. Studies have shown that ensemble learning on vertically partitioned datasets enhances the accuracy of inference results. Hence, a technique for confidential inference and training is a need, especially when the dataset is vertically partitioned. In the past, vertically partitioned datasets brought complexity and performance penalty to privacy-preserving machine learning solutions; that's where the proposed technique would bring accuracy and private gain.

3.2. Technique. A typical vertically partitioned dataset would have multiple data owners having different schemas for details about a single entity. The proposed technique uses the Secure Multi-Party Computation (SMPC) technique to train the intermediate models that privately address data owners' privacy concerns. Similarly, the intermediate model and secret are shared in SMPC clusters with multiple secure nodes such that no single node gets to know the complete model, thereby preserving the privacy of model owners. The same SMPC cluster carries out the private inference when client values are secret and shared over the same secure nodes, so a secure node gets a fraction of inference input. The inference output is generated at each secure node, and later it gets aggregated by a result aggregator, which applies differential privacy-based secure aggregation on the final inference output. The degree of privacy is controlled by 'epsilon' or privacy budget. The final inference output is later shared over the network in an encrypted format.

3.3. Architecture. As shown in Fig 3.1, the vertically partitioned data has three splits named Feature Split A (FSA), Feature Split B (FSB), and Target List. The split has different schemas where FSA and FSB have a list of feature values or x values, while the TL has the target value or y value as one of the columns. As depicted in the figure, FSA and Target List are securely trained using SMPC, which generates an intermediate model named Ma; similarly, Mb is generated by SMPC-based secure training of FSB and Target List. The application of SMPC has ensured the privacy of FSA, FSB, and Target List data owners. The model's Ma and Mb are secretly shared over another SMPC-based cluster of total N Secure Nodes, namely SN_a, SN_b, SN_c…SN_n such that M_a is divided into N multiple splits M_{aa}, M_{ab}, M_{ac}…M_{an} and shared across secure nodes. The secret share ensures the privacy of model owners, as no secure node has the complete model.

The inference input X is similarly divided across N secure nodes such that SN_a, SN_b, SN_c…SN_n gets X_a, X_b, X_c…X_n values, and it makes the inference value private too. The secure nodes compute the intermediate

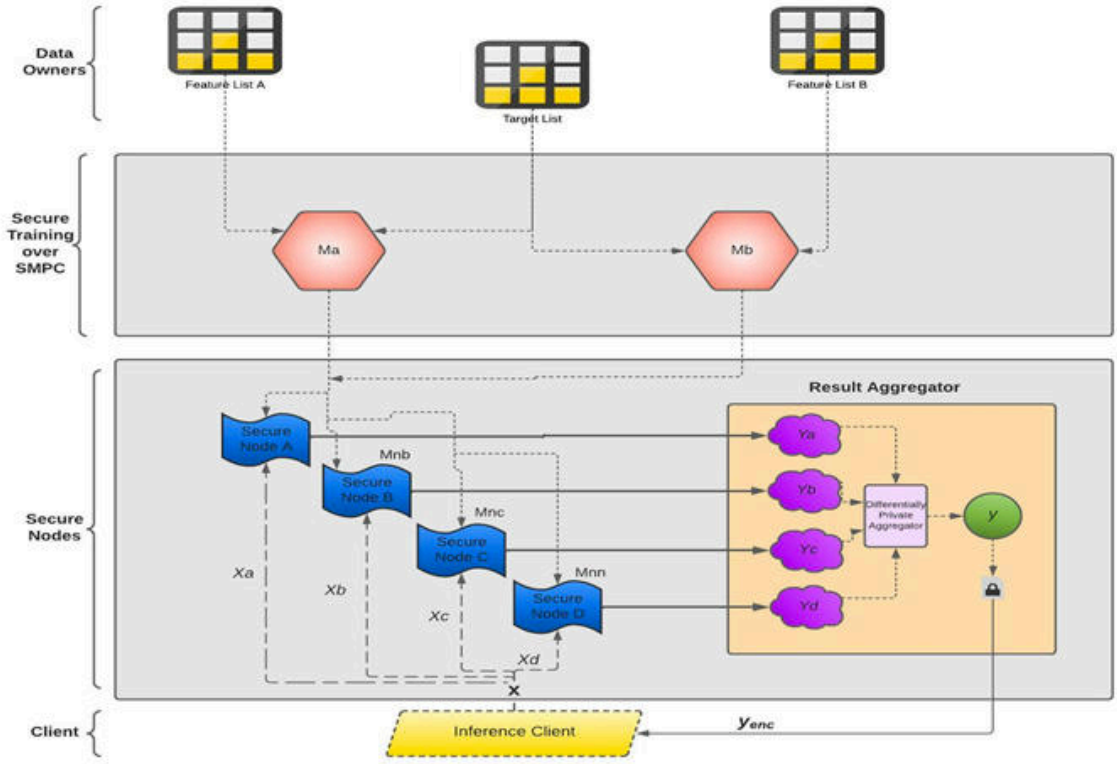

Fig. 3.1: CoTraIn-VPD Architecture

inference outputs like Y_a, Y_b, Y_c...Y_n, which is aggregated by a trusted result aggregator who applied differential privacy aggregation to arrive at the final inference output Y', which is later shared over the network in an encrypted format $Y'_{encrypt}$ to preserver network security. Nowhere the data, model, or inference input has been shared in its original form and value, additionally the collaborative computation occurs on different secure nodes make the technique privacy-preserving.

The methodology of CoTraIn-VPD can be broken down into the following steps:

1. Vertical partitioning of the data: The dataset is partitioned into two or more parts, with each part held by a different party. The partitioning is done such that each party has access to only a subset of the features (columns) of the data, while the full set of records (rows) is distributed across the parties.
2. Local model training: Each party trains a local model on its own data using standard machine learning techniques. The local model is trained on the subset of features that the party has access to, and the output is a model that predicts the target variable based on that subset of features.
3. Secure aggregation: The local models are combined in a secure and privacy-preserving way to obtain a global model that can predict the target variable based on all the features. This is achieved using techniques such as secure multi-party computation (MPC) or homomorphic encryption.
4. Fine-tuning: The global model is fine-tuned using a small amount of jointly-held data to improve its accuracy. This is done in a privacy-preserving way using techniques such as differential privacy or federated learning.

3.4. Mathematical Explanation. As per the proposed training technique, two different splits get trained into a secure multi-party computation setting simultaneously, and the same can be parallelized. Hence, the training time t_{train} is constant and does not depends on the number of vertical splits vs If K is a constant and

t_{train} is training time, we can say:

$$t_{\text{train}} \propto K$$

Similarly, the model is securely distributed to the number of secure nodes and the secure computation occurs at each secure node which later gets aggregated to form the final inference result. Although inference time would increase with secure nodes count because of the computation and aggregation complexity. However, the inference time does has no correlation with number of models participating in the SMPC cluster. Hence, if the time taken to inference is $t_{inference}$ and the secure nodes count is N_{sec_nodes}.

$$t_{\text{inference}} \propto N_{\text{sec\_}nodes}$$

The aggregation adds the statistical noise based on the privacy budget epsilon, which, too, does not impact the time taken to inference. Hence, t does not depend on epsilon.

The CoTraIn-VPD approach has several ethical implications, particularly with regards to data privacy and ownership. Below are some of the potential ethical concerns and considerations:

1. Data privacy: The use of vertically partitioned datasets introduces privacy concerns, as each party may have access to sensitive information that they are not authorized to see. The CoTraIn-VPD approach attempts to address this by limiting each party's access to only a subset of the features, but there is still a risk of sensitive information being leaked if the secure aggregation and privacy-preserving techniques are not implemented correctly.
2. Data ownership: The partitioning of the data means that each party has ownership over their subset of the data. This can lead to issues around data sharing and access, as parties may not want to share their data with others. It is important to establish clear ownership and usage rights for each party, and to ensure that consent is obtained from all parties before any data sharing takes place.
3. Bias and fairness: The use of local models can introduce bias into the global model, as each party may have their own biases and assumptions that are reflected in their local model. This can lead to unfair treatment of certain groups or individuals. It is important to carefully consider the features used by each party and to ensure that the global model is fair and unbiased.
4. Transparency and accountability: The use of secure aggregation and privacy-preserving techniques can make it difficult to understand how the global model is making its predictions. This can make it difficult to hold the parties involved accountable for any errors or biases in the model. It is important to establish clear guidelines for transparency and accountability, and to ensure that the parties involved are able to explain and justify their decisions.
5. Informed consent: In order to participate in vertically partitioned data sharing, parties must give informed consent. This means that they must be fully aware of the potential risks and benefits of sharing their data, and must understand how their data will be used and protected. It is important to ensure that all parties involved have given informed consent, and that any changes to the data sharing agreement are communicated clearly and transparently

3.5. Experimental Setup. As part of the simulation, a virtual machine on Azure cloud with Ubuntu OS was used. The machine's configuration was D2sV3, 2vCPU and RAM as 8GiB. Open-source libraries like PyTorch, PySyft, PyDP, CrypTen, and SyMPC (OpenMined) were used, and Jupiter Notebook was the development environment. The language used was Python 3.9 and dataset was Boston Housing Dataset, with parameters including no. of model's owners: 1to 4, no. of Secure Nodes: 2 to10, and a Result assembler. Pseudocode is stated below,

4. Result and Analysis. Table 4.1 shows the results of running the CoTraIn-VPD approach without differential privacy on two vertically partitioned datasets (VP1 and VP2). The results are presented for different numbers of secure nodes and different numbers of splits in the vertical partitioning. For VP1, with two splits, the inference time ranges from 3 seconds for 3 secure nodes to 11 seconds for 10 secure nodes. The mean squared error (MSE) loss for VP1 is 18.83. For VP2, with three splits, the inference time ranges from 1.42 seconds for 1 secure node to 15.85 seconds for 10 secure nodes. The MSE loss for VP2 is 14.96.

```
Function Secure_Train_Models (n_SecureNodes,n_Models, smpc_Protocol=None)
{
    # split data vertically into multiple feature and one target set based on the columns
    Data_feature_split_A = split columns wise A features
    Data_feature_split_B = split columns wise B features
    Data_feature_split_C = split columns wise C features
    Data_target_split = split columns wise target values

    # securly train each feature set with the target set
    model_A = SecureTrain (Data_feature_split_A, Data_target_split)
    model_B = SecureTrain (Data_feature_split_B, Data_target_split)
    model_C = SecureTrain (Data_feature_split_C, Data_target_split)

    # privately share the model to Differential_Private_And_Secure_Inference function
    n_Models.append(model_A)
    n_Models.append(model_B)
    n_Models.append(model_C)
}
```

```
Function Differential_Private_And_Secure_Inference(n_SecureNodes,n_Models,inference_Data,smpc_Protocol=None)
{
    # Setup the SMPC session for the computation
    if smpc_Protocol == None
        Initialize SMPC session by creating n_SecureNodes vitual machines
        and using SMPC protocol = SPDZ
    Else
        Initialize SMPC session by creating n_SecureNodes vitual machines
        and using SMPC protocol = smpc_Protocol
    Endif
    # Secret share each model to SMPC n_SecureNodes
    For each model in n_Models
        secret share model to SMPC n_SecureNodes resulting into secure model
        append secure model into n_SecureModels
    Endfor
    # Secret share inference data to SMPC n_SecureNodes
    secret share inference data x to SMPC n_SecureNodes resulting secure_Inference_Data
    # Evalute inference at each secure node and generate results
    For each secureModel in n_SecureModels
        calculate inference result with secureModel on secure_Inference_Data
        append the result secure_Results
    Endfor
    # differentially private aggregation of result into final result
    apply differential private aggregation to secure_Result to arrive at final_Result
    Encrypt the final_result to enc_Final_Result
    Return enc_Final_Result
end function
```

Fig. 3.2: Pseudo Code for CoTraIn-VP

Overall, the table shows that increasing the number of secure nodes or splits in the vertical partitioning can increase the inference time, but does not necessarily lead to a reduction in MSE loss. It is important to balance the trade-off between inference time and accuracy when selecting the optimal configuration for the CoTraIn-VPD approach. It should be noted that the results presented in this table 4.1 are without the use of differential privacy. In real-world scenarios, the use of differential privacy may be necessary to protect the privacy of the vertically partitioned datasets. The results may differ when differential privacy is applied.

Table 4.2 shows the results of running the CoTraIn-VPD approach with differential privacy on two vertically partitioned datasets (VP1 and VP2). The results are presented for different numbers of secure nodes and different numbers of splits in the vertical partitioning. For both VP1 and VP2, the inference time is consistent across all configurations, at 0.97 seconds. This is because differential privacy introduces a noise mechanism that adds random noise to the computations, leading to more consistent inference times. However, the MSE loss

Table 4.1: Results without Differential Privacy

Vertical Partitio ning	Secure Node	Inference Time	MSE Loss	Vertical Partitio ning	Inference Time	MSE Loss
VP1 (2 splits)	3	0.968101978	18.82760239	VP 2 (3 splits)	1.422831297	14.96163082
	4	1.772582531	18.82761192		2.535790205	14.96164703
	5	2.774291992	18.82762718		4.1898036	14.96165276
	6	3.980890751	18.82762527		5.812644005	14.96162415
	7	5.856841803	18.82763672		7.877051115	14.96162033
	8	7.015391827	18.82762909		10.2261157	14.96164131
	9	9.025359154	18.82760811		12.91781378	14.96164799
	10	11.13081789	18.8276062		15.84927869	14.9616251

Table 4.2: Results with Differential Privacy

Vertical Partitio ning	Secure Node	Inference Time	MSE Loss	Vertical Partitio ning	Inference Time	MSE Loss
VP1 (2 splits)	3	0.968101978	18.82760239	VP2 (3 splits)	0.968101978	18.82760239
	4	1.772582531	18.82761192		1.772582531	18.82761192
	5	2.774291992	18.82762718		2.774291992	18.82762718
	6	3.980890751	18.82762527		3.980890751	18.82762527
	7	5.856841803	18.82763672		5.856841803	18.82763672
	8	7.015391827	18.82762909		7.015391827	18.82762909
	9	9.025359154	18.82760811		9.025359154	18.82760811
	10	11.13081789	18.8276062		11.13081789	18.8276062

for both VP1 and VP2 remains the same as without differential privacy, at 18.83 and 14.96, respectively. This suggests that the use of differential privacy has not significantly impacted the accuracy of the CoTraIn-VPD approach.

Overall, the results in Table 4.2 demonstrate the effectiveness of the CoTraIn-VPD approach with differential privacy in achieving accurate predictions while preserving the privacy of the vertically partitioned datasets. It is important to note that the use of differential privacy may introduce additional computational costs and considerations, such as selecting the appropriate privacy parameters and noise mechanisms

4.1. Insights.

4.1.1. Partition vs Performance. We got stereotype time taken across various values of secure nodes using different values privacy budget (epsilon 0.2, 0.6, and 0.8). The experiment revealed the time taken for 2 splits was marginally better than 3 splits. However, the trend showed the time taken increases linearly across various privacy budgets with an increase in secure nodes. This explains that the number of splits does not create a huge performance loss across various privacy budgets. The technique does not impact the performance while keeping the inference computation private for all the stakeholders.

4.1.2. Partition vs Privacy. Including differential privacy at the result aggregation increases the privacy for inference clients, model owners, and data owners without impacting accuracy. The experiment shows that the MSE loss remains within the benchmark and does not depend on the number of splits. Including differential privacy brings randomness to the inference output but does not deviate significantly when the privacy budget is increased. The technique showcased accuracy price is a bare minimum and does not depend on the number of splits.

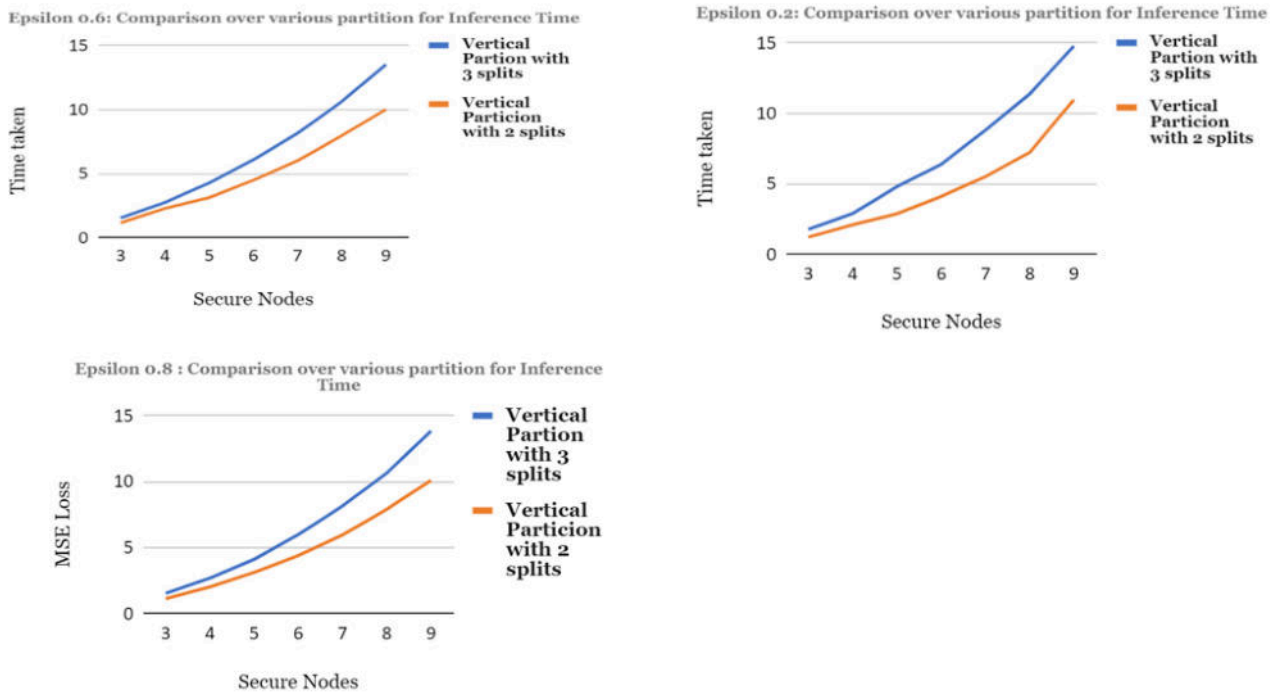

Fig. 4.1: Partition v/s Performance : (a) Epsilon=0.2 (b) Epsilon=0.6 (c) Epsilon=0.9

4.1.3. Privacy vs Accuracy across Partitions. The graph showcased the decrease in MSE loss when epsilon is increased, and this remains the observation across different splits. The technique has proved that the number of splits does not impact the accuracy of inference output. Still, differential privacy increases privacy with a negligible loss of accuracy that the privacy budget can also control. The randomness observed with the privacy budget was in line with the non-private variation of model inference.

4.1.4. Privacy vs Performance across Partitions. The graph represents the mean inference time taken across various privacy budgets is marginally higher than 2 splits. Although, the time taken remains constant across various privacy budgets, which means the technique can give a consistent performance across different values of privacy budgets.

4.1.5. Secure Nodes vs Performance. The observations align with CoInMpro's [22] finding that growing secure nodes increases processing across various nodes, resulting in a linear time increase. We tested 10 secure nodes and found a linear increase in the time taken to inference. However, this did not correlate with the initial number of partitions.

4.1.6. Secure Nodes vs Accuracy . Differential privacy injects randomness into the inference result; however, the study concluded that increasing the secure nodes does not correlate with the accuracy of the output. Across various splits, similar behaviour was observed, and accuracy remained within an acceptable deviation infused due to differential privacy.

4.2. Measure of Privacy. To measure the privacy we need to quantify amount of information reveal by the model after applying the CoTraIn-VPD technique about the training data. With increase in

information leakage the privacy assured by the technique goes down. We have measure the information leakage through Mutual Information concept of information theory, which measure the amount of information two random variable have. In theory, we can prove gain in privacy by the PPML technique, if we find feature vs target value Mutual Information decreases with CoTraIn-VPD when compared with a non-private model.

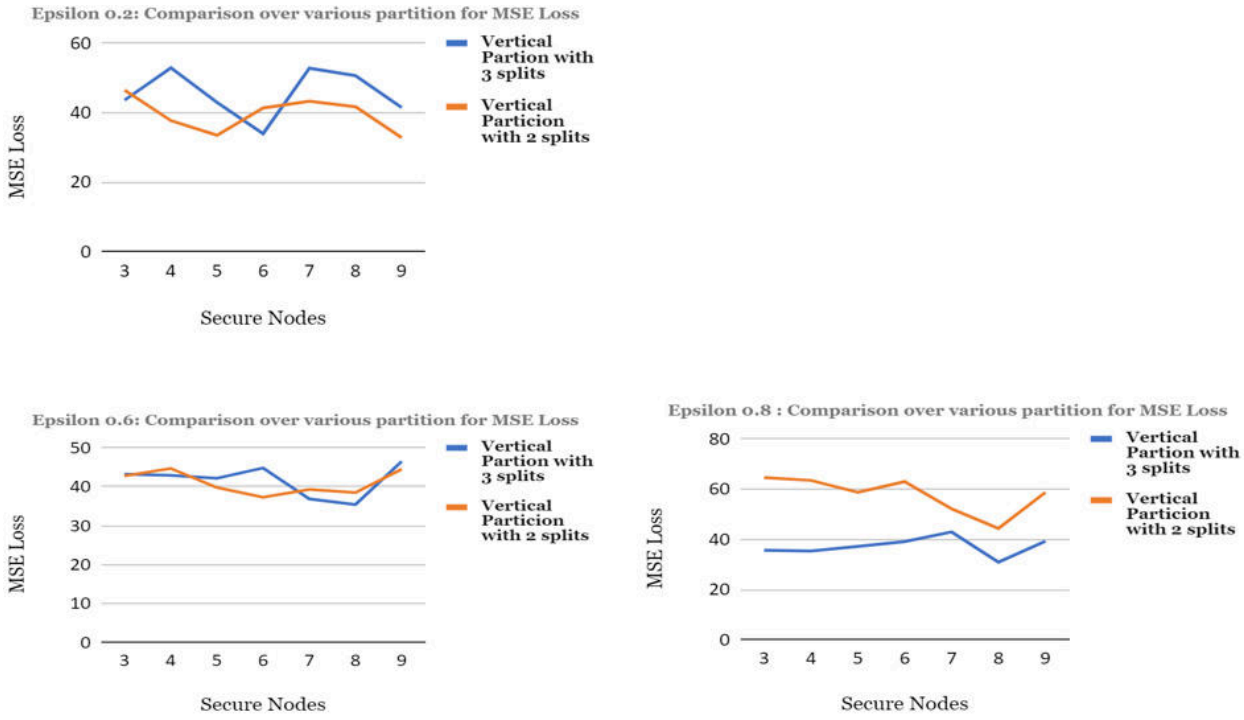

Fig. 4.2: Partition v/s Privacy: (a) Epsilon=0.2 (b) Epsilon=0.6 (c) Epsilon=0.9

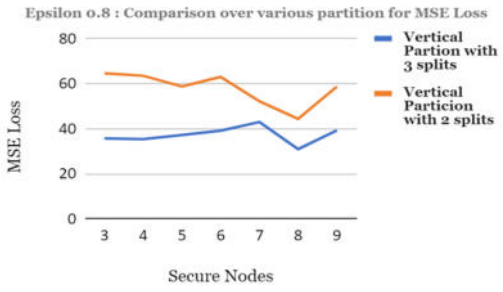

Fig. 4.3: Epsilon v/s Accuracy across Split

Figure 4.7 and 4.8 shows the MI decreased with CoTraIn-VPD across different privacy budget for different splits as compare to the non-private model denoted by privacy budget 0. It is also observed that privacy gains when we go more number of splits. KL-Divergence is another technique to find information leakage by measuring the difference between probability distribution of features and target variable.

In Figure 4.9, the experiments shown the value of KL-Divergence remains below 20 across different privacy budget over different splits.

5. Conclusion. Digitalization across sectors has opened up opportunities for easier collaboration where each sector carries a distinct set of information about individuals, customers, or firms. There is huge scope for improving productivity, efficiency, and synergy across sectors using collaborative machine learning, but is affected because of privacy concerns. There are minimal implementation-proven solutions available today that claim to solve collaborative machine learning across a vertically split dataset without paying huge penalties

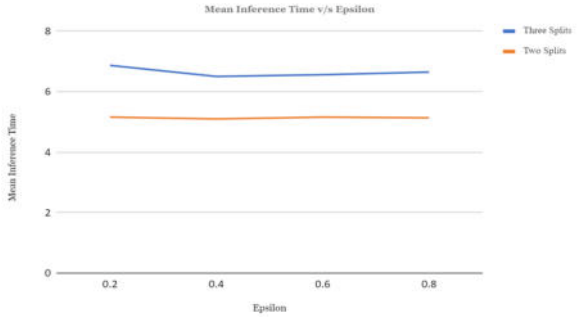

Fig. 4.4: Epsilon v/s Performance across Splits

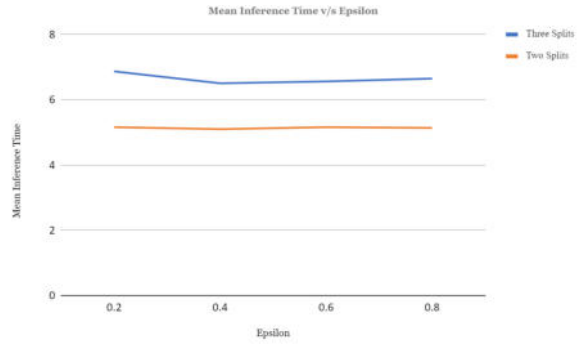

Fig. 4.5: Privacy v/s Performance across 3 splits

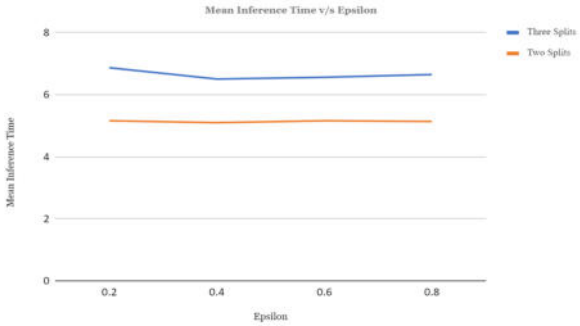

Fig. 4.6: Privacy v/s Accuracy across 3 splits

regarding accuracy and performance or measuring the privacy gain. The paper presented a technique that empirically showed privacy-preserving machine learning for a vertically partitioned dataset using secure multi-party computation techniques. The technique named CoTraIn-VPD, Confidential Training and Inference using secure multi-party computation for the vertically partitioned dataset, trains the vertical split dataset using a secure multi-party computation framework named Crypten. The code written in Python uses open-source libraries such as SyMPC for the SMPC framework, and pyDP for differential privacy features. We ran exhaustive experiments across multiple splits over Azure machine learning VMs running Ubuntu OS. The experiments showed CoTraIn-VPD technique has effectively preserved the privacy of the vertical split data owners, model

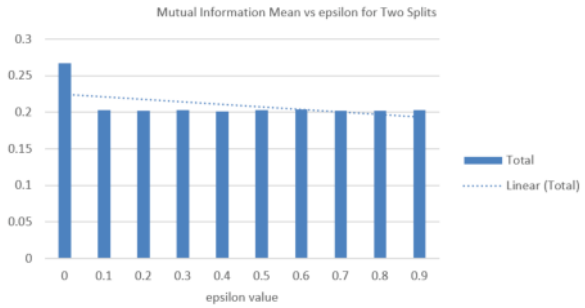

Fig. 4.7: Mutual Information across privacy budgets for two splits

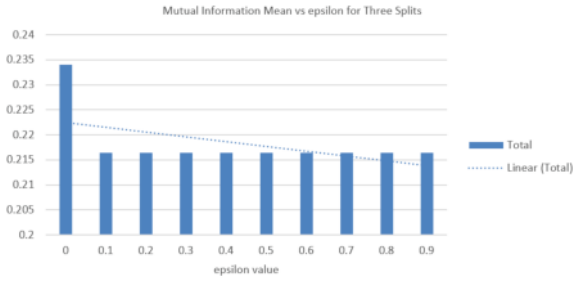

Fig. 4.8: Mutual Information across privacy budgets for three splits

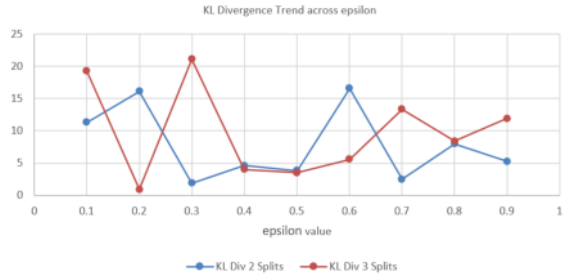

Fig. 4.9: KL-Divergence across privacy budgets and different vertical splits

owners, and inference clients with no impact on accuracy and a marginal linear impact on performance with a growth in secure nodes. The experiment proved the privacy gain using information theory metrics like Mutual Information and KL-Divergence, where information leakage decreased by applying the proposed technique. The experiments focused on two and three splits majorly but can be extended to a higher number of splits and other training and inference algorithms like logistic regression. Using a larger dataset can further extend the research to strengthen the claims of technique effectiveness.

REFERENCES

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, *Deep learning with differential privacy*, in Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, 2016, pp. 308–318.

[2] N. Agrawal, A. Shahin Shamsabadi, M. J. Kusner, and A. Gascón, *Quotient: two-party secure neural network training and prediction*, in Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 1231–1247.
[3] B. K. Beaulieu-Jones, W. Yuan, S. G. Finlayson, and Z. S. Wu, *Privacy-preserving distributed deep learning for clinical data*, arXiv preprint arXiv:1812.01484, (2018).
[4] D. Bogdanov, L. Kamm, S. Laur, and V. Sokk, *Rmind: a tool for cryptographically secure statistical analysis*, IEEE Transactions on Dependable and Secure Computing, 15 (2016), pp. 481–495.
[5] C. Bonte and F. Vercauteren, *Privacy-preserving logistic regression training*, BMC medical genomics, 11 (2018), pp. 13–21.
[6] S. Carpov, N. Gama, M. Georgieva, and J. R. Troncoso-Pastoriza, *Privacy-preserving semi-parallel logistic regression training with fully homomorphic encryption*, Cryptology ePrint Archive, (2019).
[7] M. Chase, R. Gilad-Bachrach, K. Laine, K. Lauter, and P. Rindal, *Private collaborative neural network learning*, Cryptology ePrint Archive, (2017).
[8] M. De Cock, R. Dowsley, A. C. Nascimento, D. Railsback, J. Shen, and A. Todoki, *High performance logistic regression for privacy-preserving genome analysis*, BMC Medical Genomics, 14 (2021), pp. 1–18.
[9] Y. Deng, X. Jiang, and Q. Long, *Privacy-preserving methods for vertically partitioned incomplete data*, in AMIA Annual Symposium Proceedings, vol. 2020, American Medical Informatics Association, 2020, p. 348.
[10] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans, *Secure linear regression on vertically partitioned datasets.*, IACR Cryptol. ePrint Arch., 2016 (2016), p. 892.
[11] R. Hall, S. E. Fienberg, and Y. Nardi, *Secure multiple linear regression based on homomorphic encryption*, Journal of Official Statistics, 27 (2011), pp. 669–691.
[12] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, {*GAZELLE*}*: A low latency framework for secure neural network inference*, in 27th USENIX Security Symposium (USENIX Security 18), 2018, pp. 1651–1669.
[13] Z. Li and M. Sun, *Privacy-preserving classification of personal data with fully homomorphic encryption: an application to high-quality ionospheric data prediction*, in International Conference on Machine Learning for Cyber Security, Springer, 2020, pp. 437–446.
[14] J. Liu, M. Juuti, Y. Lu, and N. Asokan, *Oblivious neural network predictions via minionn transformations*, in Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, 2017, pp. 619–631.
[15] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, *Fate: An industrial grade platform for collaborative learning with data protection*, The Journal of Machine Learning Research, 22 (2021), pp. 10320–10325.
[16] E. Makri, D. Rotaru, N. P. Smart, and F. Vercauteren, *Epic: efficient private image classification (or: Learning from the masters)*, in Topics in Cryptology–CT-RSA 2019: The Cryptographers' Track at the RSA Conference 2019, San Francisco, CA, USA, March 4–8, 2019, Proceedings, Springer, 2019, pp. 473–492.
[17] K. Mandal and G. Gong, *Privfl: Practical privacy-preserving federated regressions on high-dimensional data over mobile networks*, in Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop, 2019, pp. 57–68.
[18] P. Mohassel and Y. Zhang, *Secureml: A system for scalable privacy-preserving machine learning*, in 2017 IEEE symposium on security and privacy (SP), IEEE, 2017, pp. 19–38.
[19] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson, *Scalable private learning with pate*, arXiv preprint arXiv:1802.08908, (2018).
[20] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, *Chameleon: A hybrid secure computation framework for machine learning applications*, in Proceedings of the 2018 on Asia conference on computer and communications security, 2018, pp. 707–721.
[21] S. Shukla and G. Sadashivappa, *Secure multi-party computation protocol using asymmetric encryption*, in 2014 International Conference on Computing for Sustainable Global Development (INDIACom), IEEE, 2014, pp. 780–785.
[22] K. Tiwari, K. Bisht, and J. P. George, *Coinmpro: Confidential inference and model protection using secure multi-party computation*, in Data Science and Security: Proceedings of IDSCS 2022, Springer, 2022, pp. 1–14.
[23] K. Tiwari, S. Shukla, and J. P. George, *A systematic review of challenges and techniques of privacy-preserving machine learning*, Data Science and Security: Proceedings of IDSCS 2021, (2021), pp. 19–41.
[24] R. Yonetani, V. Naresh Boddeti, K. M. Kitani, and Y. Sato, *Privacy-preserving visual learning using doubly permuted homomorphic encryption*, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2040–2050.
[25] J. Zhang, C. Li, J. Ye, and G. Qu, *Privacy threats and protection in machine learning*, in Proceedings of the 2020 on Great Lakes Symposium on VLSI, 2020, pp. 531–536.

Edited by: Sathishkumar V E
Special issue on: Scalability and Sustainability in Distributed Sensor Networks
Received: Apr 6, 2023
Accepted: Jun 24, 2023