Secure Distributed Sparse Gaussian Process Models Using Multi-Key Homomorphic Encryption

Adil Nawaz[1], Guopeng Chen[1], Muhammad Umair Raza[1], Zahid Iqbal[2], Jianqiang Li[3], Victor C.M. Leung[1], Life Fellow, IEEE, Jie Chen[1]*

[1]College of Computer Science & Software Engineering, Shenzhen University, Shenzhen, China
[2] College of Electronics & Information Engineering, Shenzhen University, Shenzhen, China
[3] National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen, China
{adilnawaz, chenguopeng2021, umair}@email.szu.edu.cn, zahidaryan@szu.edu.cn, lijq@szu.edu.cn, vleung@ieee.org

Abstract

Distributed sparse Gaussian process (dGP) models provide an ability to achieve accurate predictive performance using data from multiple devices in a time efficient and scalable manner. The distributed computation of model, however, risks exposure of privately owned data to public manipulation. In this paper we propose a secure solution for dGP regression models using multi-key homomorphic encryption. Experimental results show that with a little sacrifice in terms of time complexity, we achieve a secure dGP model without deteriorating the predictive performance compared to traditional non-secure dGP models. We also present a practical implementation of the proposed model using 15 Nvidia Jetson Nano Developer Kit modules to simulate a real-world scenario. Thus, secure dGP model plugs the data security issues of dGP and provide a secure and trustworthy solution for multiple devices to use privately owned data for model computation in a distributed environment availing speed, scalability and robustness of dGP.

Introduction

Increase in adoption of Internet of Things (IoT) devices for various tasks has caused explosive growth in the generation of distributed data in recent years. This phenomenon has the potential to enable novel machine learning (ML) applications in domains such as smart healthcare, finance, smart homes or autonomous vehicles. However, in such applications, data is distributed among several devices separated in space, such as different healthcare institutions, financial institutions, research facilities or the end-users. Training ML models traditionally requires spatially dispersed, heterogeneous data to be collected at a centralized server, entailing several challenges like high volumes, communication complexity, secure management and processing. Additionally, competition, bureaucratic red-tape and data security laws, such as the European Union General Data Protection Regulation (EU GDPR) also restrict collection of distributed data by centralized servers for training ML models (Horvitz and Mulligan 2015). Moreover, physical and safety-critical applications, such as control systems in medical robots or autonomous vehicles, require high quality models that guarantee reliable solutions without compromising security and privacy of local data (Lederer et al. 2021).

Gaussian Process (GP) models have very good prediction accuracy and provide accurate estimates of uncertainty in prediction, hence proving to be one of the best candidates in critical applications such as healthcare or finance which require risk-averse solutions. However, achieving data security in full Gaussian process (FGP) model is expensive in terms of computation because it involves a large number of arithmetic operations with increase in data size. A secure FGP model proposed by (Fenner and Pyzer-Knapp 2020) uses fully homomorphic encryption (FHE) (Brakerski 2012) in a client-server scenario, where a GP model at server computes predictions for client using client's data. However, despite careful implementation, secure FGP model is impractical in real-world applications due to its very poor scalability and high memory and time complexity. In an approach to approximate the GP model in a distributed setting, (Chen et al. 2013) proposed dGP models computed over a cluster of computing devices thus providing access to data for model computation without the need to share it on a central server. Compared to traditional sparse GP approximation (Snelson and Ghahramani 2005; Rasmussen 2003; Seeger, Williams, and Lawrence 2003; Csató and Opper 2002), dGP models are more efficient because computation load is shared among a set of collaborating computing devices which enables application of such models in real-time. In dGP computation, each device uses its *local data* to compute a *local summary* which is a data structure comprising of a mean vector and covariance matrix comprising of the expected values and covariances among realized outputs respectively. The *local summaries* of all the devices involved in computation of dGP model are aggregated into a *global summary*, which is then used for predictions on unobserved data. This setting reduces the time and memory complexity compared to centralized full and sparse GP models. The predictive posterior distribution has lower uncertainty around the region closer to observed data points due to higher covariance among unobserved and observed points in close proximity realized data, therefore, in this setting, privacy of the local data of each device is not preserved, where an adversary can use the *local summary* structure to predict particular devices' *local data*.

*Corresponding author (email: chenjie@szu.edu.cn)

Intuitively, a third-party server can be introduced to hold *local summaries* from each device and compute a *global summary* for predictions. However, legislation like EU GDPR and threat of data privacy breach at server may hinder the devices from sharing *local summaries* with centralized server.

An intuitive solution to ensure the security of *local data* used for *local summary* computation is to encode the *local summary* in such a way that it cannot be decoded by any other device. Moreover, *global summary* computed from the encoded *local summaries* is collaboratively decoded by all the devices. However, achieving security for distributed computation model is challenging and non-trivial task. The distributed nature of data and local and global computation of the model introduces the risk of privacy leakage at multiple levels. This work proposes a method to achieve privacy in local as well as global computation of dGP model by utilizing homomorphic encryption (HE) (Gentry 2009) scheme adapted for the distributed computation scenario (Chen et al. 2019; Ma et al. 2022). The privacy of local data used in computation is preserved by leveraging HE in *Secure Distributed Sparse GP (SDS-GP)* framework. In addition to preserving privacy of the local data, the proposed scheme does not degrade the model performance in terms of predictive performance as well as computation and memory complexity. In summary, following are the main contributions of this work:

• We propose the SDS-GP framework implementing dGP algorithms with HE to perform secure computation over a network which is the first such scheme to the best of our knowledge.

• Furthermore, to provide evidence of its accuracy, scalability and efficiency in distributed computation setting and simulate real-world application scenario, we also provide a practical implementation of the proposed scheme over a set of 15 Nvidia Jetson Nano devices.

• Finally, we present empirical evidence to demonstrate the superiority of the proposed scheme over non-secure GP models.

Distributed Sparse GP Models

GPs are non-parametric Bayesian models based on the assumption that a set of multivariate random variables follow a multivariate Gaussian distribution. In FGP, a prior is placed on set of all possible functions for input domain $\mathcal{X}$. Prior distribution is Gaussian with mean μ_x and covariances $\sigma_{xx'}$ computed by a kernel function $k(\mathbf{x}, \mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. An FGP model makes use of the observed points in $(\mathcal{D}, y_\mathcal{D}) \subset \mathcal{X}$ to predict unobserved data $\mathcal{U} \subset \mathcal{X}$ using posterior mean and covariances as follows.

$$\mu_{\mathcal{U}|\mathcal{D}} \triangleq \mu_\mathcal{U} + \boldsymbol{\Sigma}_{\mathcal{U}\mathcal{D}}\boldsymbol{\Sigma}_{\mathcal{D}\mathcal{D}}^{-1}(y_\mathcal{D} - \mu_\mathcal{D}), \tag{1}$$

$$\boldsymbol{\Sigma}_{\mathcal{U}\mathcal{U}|\mathcal{D}} \triangleq \boldsymbol{\Sigma}_{\mathcal{U}\mathcal{U}} - \boldsymbol{\Sigma}_{\mathcal{U}\mathcal{D}}\boldsymbol{\Sigma}_{\mathcal{D}\mathcal{D}}^{-1}\boldsymbol{\Sigma}_{\mathcal{D}\mathcal{U}} \tag{2}$$

where $\mu_{\mathcal{U}|\mathcal{D}}$ and $\boldsymbol{\Sigma}_{\mathcal{U}\mathcal{U}|\mathcal{D}}$ denote posterior mean vector and covariance matrix respectively with $\mu_\mathcal{U}$ and $\mu_\mathcal{D}$ comprising of mean elements $\mu_\mathbf{x}$ for $\mathbf{x} \in \mathcal{U}$ and $\mathbf{x} \in \mathcal{D}$ respectively. $\boldsymbol{\Sigma}_{\mathcal{U}\mathcal{D}}$ comprises of covariance elements $\sigma_{\mathbf{x}\mathbf{x}'}$ for $\mathbf{x} \in \mathcal{U}$ and $\mathbf{x}' \in \mathcal{D}$, $\boldsymbol{\Sigma}_{\mathcal{D}\mathcal{D}}^{-1}$ with elements $\sigma_{\mathbf{x}\mathbf{x}'}$ for $\mathbf{x}, \mathbf{x}' \in \mathcal{D}$.

The FGP models however prove to be impractical in most real-world applications due to their prohibitive computation cost (i.e. $\mathcal{O}(|\mathcal{D}|^3)$ in Eqs. (1) and (2). Instead of using FGP, sparse GP models (Snelson and Ghahramani 2005) make use of a support set $\mathcal{S} \subset \mathcal{X}$ to approximate the FGP, thus decreasing the computation cost. Sparse GP approximation reduces the computation cost of FGP models, but still requires centralized computation of the model to fit data. On the other hand, dGP approximation (Chen et al. 2013) provides a way to fit the local data to compute a global sparse GP approximation in a distributed computation setting where a number of collaborating computing devices can take part in computation without need to share the local training data. The dGP models, therefore provide flexibility (in terms of number of computing devices and distance) while maintaining the performance of highly accurate GP models.

Secure Distributed Sparse GP Models

Security Analysis of Distributed Sparse GP

Distributed sparse GP model is an approximation of FGP where several devices collaborate to compute a global summary which is used for making the predictions. Each collaborating device acquires local data, e.g., client records of a financial organization to fit a dGP model. Mathematically, local data for each device is defined as follows.

Definition 1 (Local Data)**.** Local data at each device i comprises of $(\mathcal{D}_i, y_{\mathcal{D}_i})$, such that $\mathcal{D}_i \subset \mathcal{X}$, $\mathcal{X}$ being the input domain and $y_{\mathcal{D}_i}$ corresponds to the realized outputs in $\mathcal{D}_i$.

A common support set $\mathcal{S} \subset \mathcal{X}$ for $|\mathcal{S}| \ll |\mathcal{D}|$ is selected prior to local summary computation. Each device uses the support set and local data to compute a local summary which is defined as follows.

Definition 2 (Local Summary (Chen et al. 2013))**.** Given local data $(\mathcal{D}_i, y_{\mathcal{D}_i})$ and support set $\mathcal{S}$, each device computes a local summary $\mathbf{L}_i \triangleq (\dot{\mathbf{m}}_\mathcal{S}^{(i)}, \dot{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{(i)})$ comprising of a mean vector $\dot{\mathbf{m}}_\mathcal{S}^{(i)}$ and a covariance matrix $\dot{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{(i)}$.

$$\dot{\mathbf{m}}_\mathcal{S}^{(i)} \triangleq \boldsymbol{\Sigma}_{\mathcal{S}\mathcal{D}_i}\boldsymbol{\Sigma}_{\mathcal{D}_i\mathcal{D}_i|\mathcal{S}}^{-1}\left(y_{\mathcal{D}_i} - \mu_{\mathcal{D}_i}\right), \tag{3}$$

$$\dot{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{(i)} \triangleq \boldsymbol{\Sigma}_{\mathcal{S}\mathcal{D}_i}\boldsymbol{\Sigma}_{\mathcal{D}_i\mathcal{D}_i|\mathcal{S}}^{-1}\boldsymbol{\Sigma}_{\mathcal{D}_i\mathcal{S}} \tag{4}$$

where $\mu_{\mathcal{D}_i}$ comprises of the expected values of outputs in $\mathcal{D}_i$, $\boldsymbol{\Sigma}_{\mathcal{D}_i\mathcal{D}_i|\mathcal{S}}^{-1}$ is defined in similar manner as in Eqs. (1) and (2) and $\boldsymbol{\Sigma}_{\mathcal{S}\mathcal{D}_i}$ is transpose of $\boldsymbol{\Sigma}_{\mathcal{D}_i\mathcal{S}}$.

The local summary is independent of $y_\mathcal{S}$ as evident from Eqs. (3) and (4), therefore $\mathcal{S}$ can be predefined based on domain knowledge prior to observing the $\mathcal{D}_i$ and shared with the collaborating devices. The server computes global summary using the local summaries of all devices as follows.

Definition 3 (Global Summary (Chen et al. 2013))**.** Given the support set $\mathcal{S}$ and the local summaries $\mathbf{L}_i$ from collaborating devices, the server computes global summary $\mathbf{G}$ such that $\mathbf{G} \triangleq (\overline{\mathbf{m}}_\mathcal{S}, \overline{\mathbf{K}}_{\mathcal{S}\mathcal{S}})$ for

$$\overline{\mathbf{m}}_\mathcal{S} \triangleq \sum_{i=1}^{M} \dot{\mathbf{m}}_\mathcal{S}^{(i)}, \quad \overline{\mathbf{K}}_{\mathcal{S}\mathcal{S}} \triangleq \boldsymbol{\Sigma}_{\mathcal{S}\mathcal{S}} + \sum_{i=1}^{M} \dot{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{(i)} \tag{5}$$

where $\dot{\mathbf{m}}_\mathcal{S}^{(i)}$ and $\dot{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{(i)}$ are the elements of local summary $\mathbf{L}_i$ as defined in Eqs. (3) and (4) and M is the number of collaborating computing devices (Machines).

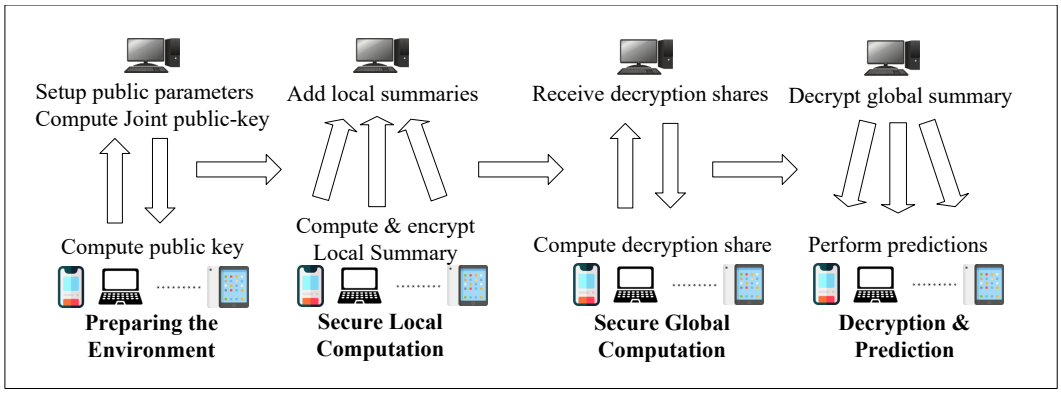


Figure 1: Schematic diagram of Secure Distributed Sparse GP (SDS-GP) framework involving four stages.

The global summary $\mathbf{G}$ is then used to make predictions on unobserved points using distributed PITC (dPITC) algorithm as follows.

Definition 4 (dPITC (Chen et al. 2013))**.** Given global summary $\mathbf{G}$, support set $\mathcal{S}$, predictive Gaussian distribution for dPITC is $\mathcal{N}(\widehat{\mathbf{m}}_{\mathcal{U}_i}, \widehat{\mathbf{K}}_{\mathcal{U}_i\mathcal{U}_i})$ for unobserved points in $\mathcal{U}_i \subset \mathcal{X}$ where

$$\widehat{\mathbf{m}}_{\mathcal{U}_i} \triangleq \mu_{\mathcal{U}_i} + \boldsymbol{\Sigma}_{\mathcal{U}_i\mathcal{S}}\overline{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{-1}\overline{\mathbf{m}}_{\mathcal{S}}, \tag{6}$$

$$\widehat{\mathbf{K}}_{\mathcal{U}_i\mathcal{U}_i} \triangleq \boldsymbol{\Sigma}_{\mathcal{U}_i\mathcal{U}_i} - \boldsymbol{\Sigma}_{\mathcal{U}_i\mathcal{S}}\left(\boldsymbol{\Sigma}_{\mathcal{S}\mathcal{S}}^{-1} - \overline{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{-1}\right)\boldsymbol{\Sigma}_{\mathcal{S}\mathcal{U}_i} \tag{7}$$

where $\mu_{\mathcal{U}_i}$, $\boldsymbol{\Sigma}_{\mathcal{U}_i\mathcal{U}_i}$, $\boldsymbol{\Sigma}_{\mathcal{S}\mathcal{S}}^{-1}$ and $\boldsymbol{\Sigma}_{\mathcal{U}_i\mathcal{S}}$ are defined as in Eqs. (1) and (2) and $\mathbf{K}_{\mathcal{S}\mathcal{U}_i}$ is transpose of $\mathbf{K}_{\mathcal{U}_i\mathcal{S}}$, whereas $\overline{\mathbf{K}}_{\mathcal{S}\mathcal{S}}^{-1}$ and $\overline{\mathbf{m}}_{\mathcal{S}}$ are defined in Eq. (5).

In posterior, the uncertainty around the observed data points is very low as compared to the uncertainty around the unobserved data. Mathematically, since $\widehat{\mathbf{m}}_{\mathcal{U}_i}$ is dependent on $\overline{\mathbf{m}}_{\mathcal{S}}$ which is obtained by aggregating the local summaries $\dot{\mathbf{m}}_{\mathcal{S}}^{(i)}$ summarizing the local data $\mathcal{D}_i$ hence the mean function passes through the data points in $y_{\mathcal{D}_i}$. Similarly, covariance among the points closer to the points in $\mathcal{D}_i$ is higher hence results in low uncertainty around observed data points.

Considering a scenario where a user with device i shares its local summary $\mathbf{L}_i$ with another user having device j. In this case, if j provided with $\mathbf{L}_i$ computes a global summary $\mathbf{G}$ from $\mathbf{L}_i$ and then uses $\mathbf{G}$ to make predictions on unobserved (test) points, it may reveal the local training points of i. For example, two users collaborate to compute a model to predict about prevalence of a disease in a particular area. The user with device j, given $\mathbf{L}_i$, can infer about some patients included in $\mathcal{D}_i$ by using some publicly available information of that area. Since the data points which belong to the local training data $\mathcal{D}_i$ have low uncertainty, therefore can be predicted with higher confidence. In such case, the privacy of local user's data is not ensured. Alternatively, if a third-party server is introduced to compute the global summary $\mathbf{G}$, instead of all devices computing $\mathbf{G}$ independently. The privacy of the local user data is only ensured if the third-party server is a trusted party. However, such a setting poses risk of privacy leakage for all the devices involved in the computation if integrity of the third-party server is compromised.

Secure Distributed Sparse GP Framework

Privacy leakage issue in the dGP models is resolved by using a multi-party HE-based scheme (Cheon et al. 2017; Ma et al. 2022) for the distributed computation scenario. Distributed computation in dGP approximation involves different tasks to be completed by the collaborating computing devices, therefore, the whole process of SDS-GP framework is decomposed into four stages as shown in Figure 1.

Preparing the Environment: Our scheme assumes that the homomorphic evaluation of functions in the encrypted form is secure because of the hardness of underlying ring learning with error problem, thus providing security for the training data by allowing us to evaluate functions without the need for an intermediate decryption. The security parameters for computation of SDS-GP framework are setup as follows.

Definition 5 (Setup HE Parameters)**.** For a positive integer q, set N to be a power of two for security and simplicity, a positive integer h and a positive real value for σ. Generate a random vector $\mathbf{a} \in \mathcal{R}_q^N$ with the elements drawn from $\mathcal{R}_q$ uniformly. Output public parameter $\mathscr{P} \triangleq (\mathbf{p}, \mathbf{a})$, where $\mathbf{p} \triangleq (N, q, \sigma, h)$.

Public parameter $\mathscr{P}$ is same for all collaborating computing devices in SDS-GP framework. Mathematical constructions for the ring and the distributions for implementing HE are as follows.

- $\Phi_K(X)$ is the K^{th} cyclotomic polynomial of degree $N = \varphi(K)$ where $\varphi(K) = |\{p : 1 \leq p \leq K \wedge \gcd(p, K) = 1\}|$.
- $\mathcal{R} = \mathbb{Z}[X]/\Phi_K(X)$ is ring of integers where ring comprises of polynomials $\mathbb{Z}[X]$, with integer coefficients modulo an ideal generated by $\Phi_K(X)$.
- $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$ is the quotient ring of $\mathcal{R}$ by ideal generated by a positive integer q, so that $\mathcal{R}_q$ comprises of residue classes of $\mathcal{R}$ modulo q.
- $\mathcal{G}(0, \sigma^2)$ denotes discrete Gaussian distribution with variance σ^2 for $\sigma > 0$. During the encryption, noise

$\mathbf{e} \sim \mathcal{G}(0, \sigma^2)$ in $\mathbb{Z}^N$ is added for *ciphertext* indistinguishability.

- $\mathcal{K}(h)$ is the set comprising of signed binary vectors in $\{0, \pm 1\}^N$ of Hamming weight h, where, hamming weight is defined as the count of non-zero entries in a vector. Each device chooses its *secret key* from the set $\mathcal{K}(h)$.
- Categorical distribution $\mathcal{E}(\rho)$ for $0 \leq \rho \leq 1$ draw elements in a binary vector $\{0, \pm 1\}^N$, where probability of drawing $+1$ or -1 is $\rho/2$ and $1 - \rho$ for 0.

After setting up security parameter $\mathscr{P}$, all the devices compute a pair of public and secret key for encryption and decryption respectively as follows.

Definition 6 (Local Key Generation)**.** Given $\mathscr{P}$ each device i generates a pair of public and secret keys $\left(\mathbf{b}^{(i)}, \mathbf{s}^{(i)}\right)$ where $\mathbf{b}^{(i)} = \left(-\mathbf{s}^{(i)} \cdot \mathbf{a} + \mathbf{e}_0^{(i)}\right) \bmod(\mathrm{q})$, for $\mathbf{s}^{(i)} \in \mathcal{K}(h)$, $\mathbf{a} \in \mathscr{P}$, $\mathbf{e}_0^{(i)} \sim \mathcal{G}(0, \sigma^2)$ and $\bmod(\mathrm{q})$ denotes arithmetic modulus operation with q as modulus.

Homomorphic computation property in the distributed computation scenario is ensured only when all the collaborating devices encrypt their local data using common encryption key. Therefore, we generate a global public key for encryption of the local summaries. Global public key is the sum of the public keys generated by each device as follows.

Definition 7 (Global Key Generation)**.** Given all the public keys $\mathbf{b}^{(i)}$, server generates a global public key $\overline{\mathbf{b}}$ for M devices such that, $\overline{\mathbf{b}} = \sum_{i=1}^{M} \mathbf{b}^{(i)} \mod(\mathrm{q})$

We assume an Honest but Curious (HbC) server which doesn't interfere in the global key generation process. However, a malicious server can generate its own public key as the global public key and send it to the devices to decrypt local summaries. In such a threat model, all the devices broadcast their respective public key to the collaborating devices and generate global public key in a decentralized manner thus eliminating the security risk posed by the malicious server.

Secure Local Computation: The secure local computation starts with the selection of an informative support set at the server which is shared with all the devices. All of the collaborating devices compute a local summary using their respective local data and the common support set (see **Definition 2**). Subsequently, the local summary is encrypted to obtain a secure local summary as follows.

Definition 8 (Secure Local Data Summarization)**.** Given local summary $\mathbf{L}_i \triangleq \left(\mathbf{m}_{\mathcal{S}}^{(i)}, \mathbf{K}_{\mathcal{S}\mathcal{S}}^{(i)}\right) \in \mathbb{R}^{|\mathcal{S}| \times (|\mathcal{S}|+1)}$, each device i encodes all the elements of local summary $[\mathbf{L}_i]_{jk} \in \mathbb{R}$ into a plaintext polynomial $\mathbf{t}^{(i)} \in \mathcal{R}$ which is encrypted into secure local summary $\mathbf{T}^{(i)} \triangleq \left(\mathbf{t_0}^{(i)}, \mathbf{t_1}^{(i)}\right) \in \mathcal{R}_q^2$ with

$$\begin{aligned} \mathbf{t_0}^{(i)} &\triangleq \mathbf{v}^{(i)} \cdot \overline{\mathbf{b}} + \mathbf{t}^{(i)} + \mathbf{e_0}^{(i)} \bmod(\mathrm{q}), \\ \mathbf{t_1}^{(i)} &\triangleq \mathbf{v}^{(i)} \cdot \mathbf{a} + \mathbf{e_1}^{(i)} \mod(\mathrm{q}) \end{aligned} \tag{8}$$

where $\mathbf{t_0}^{(i)}, \mathbf{t_1}^{(i)} \in \mathcal{R}_q$, whereas $\mathbf{v}^{(i)} \sim \mathcal{E}(\rho)$ and $\mathbf{e_j}^{(i)} \sim \mathcal{G}(0, \sigma^2)$. Encoding $(\mathrm{Ecd}(\cdot; \Delta_s))$ of a message into the plaintext is elaborated in (Cheon et al. 2017).

Lemma 9 (Encryption Noise (Cheon et al. 2017))**.** *If* $\mathrm{Ecd}(\cdot; \Delta_s)$ *and* $\mathrm{Enc}(\cdot)$ *denote encoding and encryption respectively and* $[\mathbf{T}^{(i)}]_{jk} \leftarrow \mathrm{Enc}([\mathbf{t}^{(i)}]_{jk})$ *and* $[\mathbf{t}^{(i)}]_{jk} \leftarrow \mathrm{Ecd}([\mathbf{L}_i]_{jk}; \Delta_s)$ *for* $\Delta_s > N + 2B_{enc}$*, where* Δ_s *is scaling factor for encoding,* B_{enc} *is the noise bound for encryption given by,* $B_{enc} = 8\sqrt{2}\sigma N + 6\sigma\sqrt{N} + 16\sigma\sqrt{hN}$*, then,* $\mathrm{Dcd}(\mathrm{Dec}([\mathbf{T}^{(i)}]_{jk})) = [\mathbf{L}_i]_{jk}$ *where* $\mathrm{Dcd}(.)$ *and* $\mathrm{Dec}(.)$ *refer to decoding and decryption respectively.*

Secure Global Computation: Server receives secure local summary $\mathbf{T}^{(i)}$ from each device i and computes a secure global summary as follows:

Lemma 10 (Secure Global Summary)**.** *Given secure local summaries* $\mathbf{T}^{(i)}$ *from each device, server computes the secure global summary* $\mathbf{G}' \triangleq (\mathbf{G_0}, \mathbf{G_1})$ *with noise bounded by* $\sum_{i=1}^{M} B_{enc}^{(i)}$*, where* $\mathbf{G_0} = \sum_{i=1}^{M} \mathbf{t_0}^{(i)} \mod(\mathrm{q})$ *and* $\mathbf{G_1} = \sum_{i=1}^{M} \mathbf{t_1}^{(i)} \mod(\mathrm{q})$.

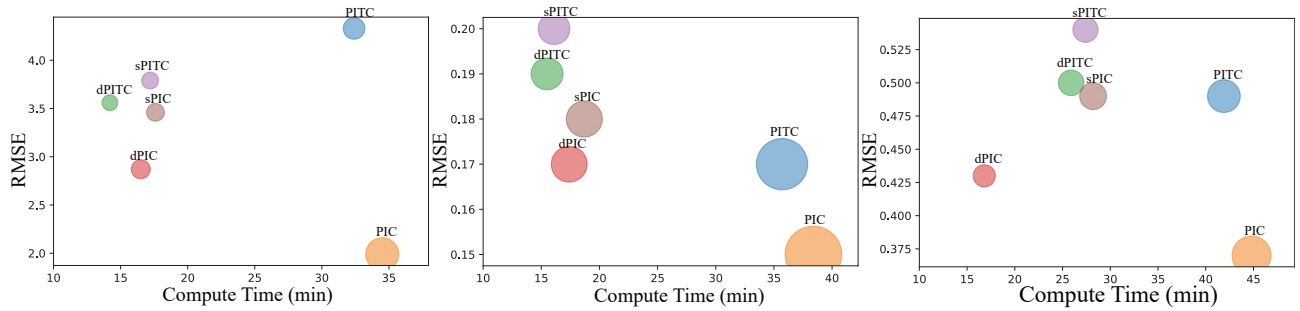
Subsequent to the computation of $\mathbf{G}$, server initiates the distributed decryption where each device collaborates with the server to compute the decryption of the secure global summary.

Decryption & Distributed Predictions: The collaborating computing devices compute the decryption of secure global summary $\mathbf{G}'$ in a collaborative decryption process. Subsequent to successful decryption of the global summary each device uses it to make predictions on its local test data.

Theorem 11 (Decryption & Distributed Prediction)**.** *Given secure global summary* $\mathbf{G}'$*, server computes the full decryption of the global summary* $\tilde{\mathbf{G}} \triangleq \mathbf{G_0} + \Gamma$ *for* $\Gamma \triangleq \sum_{i=1}^{M} \Gamma^{(i)} \bmod(\mathrm{q})$*, where* $\Gamma^{(i)} = \mathbf{s}^{(i)} \cdot \mathbf{G_1} + \bar{\mathbf{e}} \mod(\mathrm{q})$ *is the decryption share computed independently by each device, then* $\tilde{\mathbf{G}} = \mathbf{G} \iff \Delta_s > N + 2B$ *for* $B = \sum_{i=1}^{M} B_{enc}^{(i)}$*.*

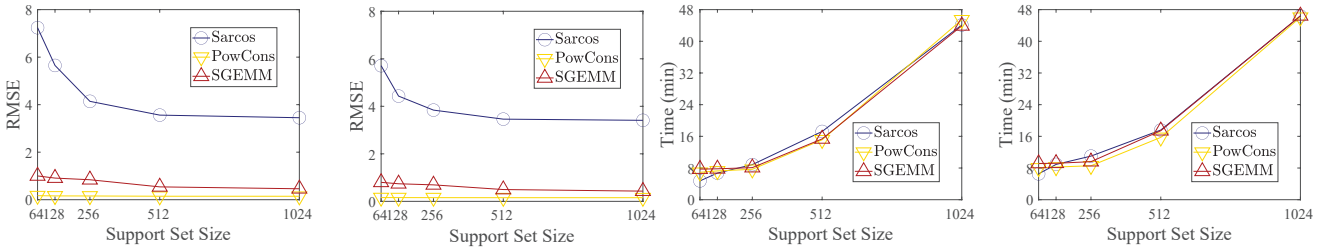
Each device i *given decrypt global summary* $\tilde{\mathbf{G}}$*, unobserved data* $\mathcal{U}_i$*, support set* $\mathcal{S}$ *and local data* $(\mathcal{D}_i, \mathbf{y}_{\mathcal{D}_i})$ *computes predictive mean and covariances for secure distributed sparse PITC (sPITC) as defined in Eqs.* (6) *and* (7) *(see* Definition 4*) and secure distributed sparse PIC (sPIC) (Chen et al. 2013).*

This setting resolves the privacy issue in sparse dGP approximation. Firstly, since the local summary of each device is encrypted and it can not be decrypted without the secret key of that particular device, hence all other devices working in collaboration with the server are unable to decrypt a particular local summary and use it for the analysis and attack. Secondly, since decryption of the global summary requires the secret keys of all the collaborating devices, hence the server is unable compute the global summary of $M - 1$ devices and use it to compute the local summary of one device, thus ensuring the confidentiality of each of the collaborating devices in a malicious threat model. Despite these security guarantees, collaboratively decrypted global summary can still reveal some information about the dGP model, however, the identity of the local user is kept anonymous thus ensuring local user's privacy. Complete immunity from the inference attack will be considered in an extension of the SDS-GP framework.



(a) Sarcos (b) PowCons (c) SGEMM

Figure 2: Computation time and RMSE comparison for different GP models on each of the datasets.



(a) sPITC (b) sPIC (c) sPITC (d) sPIC

Figure 3: RMSE and compute times for varying size of support set for SDS-GP framework.

Time Complexity of Secure Distributed Sparse GP: The additional computation cost for SDS-GP models come from the homomorphic encryption, addition and decryption algorithms. However, since all the algorithms involve operations on polynomials, hence only incur polynomial time increase in computation complexity. The encryption and decryption entails computation overload of order $\mathcal{O}(N\log(N))$. The addition operation simply adds the values in corresponding ciphertexts, hence only adds $\mathcal{O}(N)$ to the computation cost.

Experiments and Results

Implementation

In this section, we present the implementation of the SDS-GP framework. The source code for the framework is developed in C++ with HE implementation based on HEAAN V2.1.0. HE operations involve computation on big numbers and modulo arithmetic which is implemented using NTL 11.5.1 built in conjunction with GMP 6.2.1. All experiments are implemented using 15 Nvidia Jetson Nano 4GB developer kit modules as the client nodes and an Intel(R) Core(TM) i5-5500U CPU @ 2.40GHz with 8GB RAM as a server connected via a 10/100Mbps standard Ethernet WLAN network. The parameters for HE, $N = 2^{16}$, $q = 800$, $h = 64$, $\rho = 0.5$ and $\sigma = 3.2$. In all the experiments, after standardizing the datasets, we split them into training and test sets, where test set in each case comprises of 10% of the whole dataset. The datasets are modelled as GP specified by the squared exponential kernel $k(u, u')$ such that, $k(u, u') \triangleq \sigma_a^2 \exp(-\frac{1}{2}\sum_{i=1}^{d}(\frac{u_i - u_i'^{\,2}}{\ell_i}) + \sigma_p^2\delta_{uu'}$, where u_i, u_i' denote the i-th feature(s) of d-dimensional input vector(s) $\mathbf{u}, \mathbf{u}'$ respectively. σ_a, σ_p and l_i are the hyperparameters representing function variance, noise variance and length-scales respectively. $\delta_{uu'}$ is the Kronecker delta which is 1 if $\mathbf{u} = \mathbf{u}'$ and 0 otherwise. Hyperparameter values are learned by Maximum Likelihood Estimation (MLE) (Rasmussen 2003).

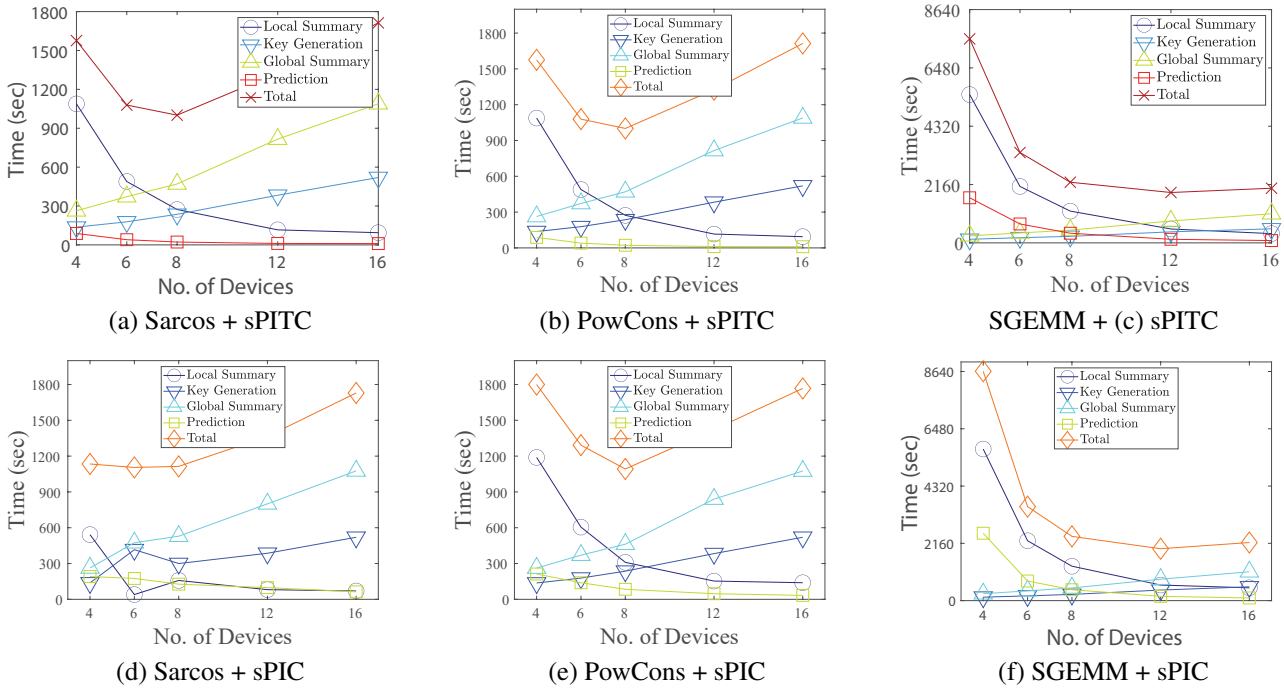
Datasets and Evaluation Metrics

The proposed scheme is evaluated on the three publicly available regression datasets, i.e, Sarcos (Vijayakumar and Schaal 2000), Power Consumption (PowCons) (Salam and Hibaoui 2018) and SGEMM GPU kernel performance (Nugteren and Codreanu 2015) dataset.

We use root mean square error (RMSE) as the evaluation metric for both sPITC and sPIC algorithms, where, $\text{RMSE} \triangleq \sqrt{|\mathcal{U}_i|^{-1}\sum_{\mathbf{x}\in\mathcal{U}_i}(y_x - \widehat{m}_{\mathcal{U}_i})^2}$ where $\widehat{m}_{\mathcal{U}_i}$ is defined in Eq. (6) and $\mathcal{U}_i$ is unobserved data of ith device. We also analyze the performance of the proposed scheme for computation and communication complexity with respect to varying the size of data and number of collaborating devices and compare the results against non-secure GP models.

Results and Discussion

Accuracy: Predictive accuracy is one of the most important metrics in ML tasks, especially when applied in the critical decision making systems. Security of the encrypted global summary depends on the noise added during the computation of ciphertext. Since noise is added to the plaintext as part of the message, we conduct an experiment to analyse



(a) Sarcos + sPITC (b) PowCons + sPITC SGEMM + (c) sPITC

(d) Sarcos + sPIC (e) PowCons + sPIC (f) SGEMM + sPIC

Figure 4: Variation in computation time by varying number of collaborating devices

the accuracy of the secure global summary vs non-secure global summary. The maximum error, i.e., $\max\left(\tilde{\mathbf{G}}-\mathbf{G}\right)$ for $\overline{\mathbf{m}}_{\mathcal{S}}$ and $\overline{\mathbf{K}}_{\mathcal{SS}}$ is 5.29×10^{-7} and 1.45×10^{-11}, respectively. We also compute the mean error, i.e., the overall error introduced by the encoding and the encryption operations in SDS-GP for secure global summary computation. The mean error for global covariance matrix $\overline{\mathbf{K}}_{\mathcal{SS}}$ is 2.89×10^{-7} and for the global mean vector $\overline{\mathbf{m}}_{\mathcal{S}}$ is 4.16×10^{-12} which is negligible. The error added to the local summary during encryption and homomorphically evaluated global summary can be considered as the rounding off error which occurs during the real-value arithmetic.

Moreover, error performance of SDS-GP is similar to the traditional non-secure dGP and FGP models. It is evident from the error performance of the proposed scheme, as illustrated in Figure 2. The RMSE of the proposed scheme for all three datasets is similar to traditional non-secure dGP models.

Computation Time vs Data Size: Size of the support set plays a critical role in performance of the SDS-GP framework. Larger size of the support set improves the accuracy of the models, however due to increase in amount of the data, the communication cost increases, hence increasing the overall computation time. Therefore, it is important to select an adequate size of the support set so as to minimize the communication cost while maintaining high accuracy. To this end we perform a number of experiments varying the size of the support set $|S|$. As evident from the results shown in Figure 3, the large $|\mathcal{S}|$ improves the accuracy while also increasing the time requirement and vice-versa. It is evident from the reported results that the computation time of sPITC(sPIC) scales linearly with the increase in size of data. It is consistent with the results previously reported by (Chen et al. 2012, 2013) for the non-secure dGP models.

Moreover, Since SDS-GP scheme comprises of the polynomial time algorithms as stated in Section , the end-to-end compute time does not increase significantly as compared non-secure dGP models. Figure 2 gives the end-to-end computation times for all the datasets which are comparable to the compute times of non-secure dGP methods.

Computation Time vs No. of Devices: Number of collaborating computing devices affect the computation time of SDS-GP. As shown in Figure 4, the computation of local summary largely affects the overall computation time. Less number of devices having large amount of data (as shown in Figure 4*(SGEMM+sPITC)* and Figure 4*(SGEMM+sPIC)*) the increase in overall computation time is significant which decrease with increasing the number of devices. However, for small datasets, the computation times for increasing number of devices first decreases because computation time for local summary decreases and then increases for further increasing the no. of devices because of additional communication time for global summary computation.

Communication Time vs Data Size & No. of Devices: Since the collaborating computing devices communicate with each other and the server during the secure model computation hence communication time is also an important metric to measure the performance and efficiency of our scheme. The communication between the devices only occurs during the key generation and secure global summary

computation stage of the secure model computation hence we see a linear increase in the communication time while increasing the number of devices as shown in Figure 4. However, on the other hand, increasing the data size does not affect the overall communication overhead in the model computation because the key generation stage is independent of the size of the data on each device and so is the global summary matrix which remains constant in size for a particular computation session. Hence the communication time remains constant for increase in the data on each device.

Related Works

The tremendous growth of data in terms of speed and volume have made it inevitable for most of the machine learning applications to adopt the distributed computation approach. The distributed machine learning (dML) however suffers from the privacy threats from both inside i.e., collaborating computing devices as well as outside i.e., from outside adversaries (Liu et al. 2022). Local data of the collaborating device is prone to privacy leakage in a distributed learning setting through its contribution in global model computation (Yao 1982). There are several techniques that ensure the security of local data in distributed ML computation including the secure multiparty computation (SMPC), differential privace (DP) and HE based secure computation models. SMPC involves secret sharing among parties while the computation of a function in a collaborative manner, however, completely secure multiparty computation requires complicated computation protocols which are achieved at cost of the reduced efficiency (Yang et al. 2019). (Mohassel and Rindal 2018) proposed a method to achieve data security under assumption of the honest & the semi-honest majority, complete security however is not considered against an honest-but-curious majority in the collaborative computation. Differential privacy is another method which uses "data-anonymisation" techniques to achieve data security in the collaborative model computation. In the prevalent DP methods, noise is added to the data in such a manner that a third party is unable to distinguish an individual data point, however, in such a setting the original data can not be restored and hence efficiency is sacrificed for achieving the security (Dwork 2008; Melis et al. 2019). Trade-off between the privacy protection and the accuracy of the model reduces the quality of the model outcomes which is not desirable in the critical applications (Naseri, Hayes, and De Cristofaro 2022).

On the other hand, strength of privacy-preservation in HE based dML methods is dependent on the underlying cryptographic scheme, therefore the models based on the robust cryptographic schemes ensure strong security guarantees without deteriorating the overall model accuracy. Homomorphic property of HE is only determined if the ciphertexts are evaluated using a common encryption key, thus sharing of the secret key determines the threat model in distributed computation. In one setting where the secret key is only known to the collaborating devices, server is unable to compromise the local data privacy (Tang et al. 2019; Xu et al. 2021; Zhang et al. 2020; Zhou et al. 2020; Chen, Li, and Miyazaki 2021; Phong et al. 2018). However, the security of the local data is dependent on the assumption that collaborating computing devices are honest and non-colluding. Moreover, server can also gain access to the secret key by creating a 'fake user' and decrypt the local data. (Phong et al. 2018) proposed a scheme that makes use of additive HE to collaboratively compute a deep learning model. However, this scheme suffers a serious drawback, i.e., all the devices use a common decryption key, therefore a strongly secure communication between the collaborating devices and the cloud server is needed to protect the privacy of the local data. In another setting where the secret key is not known to any of the collaborating devices except the server (Mandal and Gong 2019; Zhu et al. 2021), such a setting assures the privacy of the local devices' data only if the server is honest. This scenario is particularly applicable in the applications where an institution needs to improve model performance while protecting the model privacy. Since the server knows the secret key in this setting, therefore it does not ensure the privacy of the local users data/models. Some methods achieve privacy for local devices in this setting by adopting different privacy preserving methods for the local devices, such methods including SecAgg (Mandal and Gong 2019), PIVODL (Zhu et al. 2021) etc., require users to evaluate the local data in an encrypted form hence increasing the computation, communication and the memory complexity. In another setting, the secret key is shared among a number of users, hence the server requires to collude with all the devices having share of the secret key to be able to breach the local data privacy. This improves the security guarantee for the first setting at the expense of an increased communication cost. These schemes also incur extra computation cost where the users have to evaluate complex functions on the ciphertexts hence rendering them impractical in many applications requiring real-time performance. Other privacy preserving ML applications (Yuan and Yu 2014) use cloud based model where the users encrypt the raw training data using a public encryption key and the powerful cloud computing server performs the computation on data in encrypted form to achieve a converged model, however, it is prohibitive in many practical real-time application scenarios due to its financial cost.

Conclusion

This work presents a multi-key HE-based distributed sparse GP regression which breaks the process of model computation into stages and perform HE only where the data is shared among the devices. We also demonstrate the practical implementation of the scheme which shows that efficient and accurate predictive model can be computed using devices with the limited computational resources without compromising the security and integrity of the local data. The experimental data reveals that the secure distributed sparse GP approximation achieves better performance in terms of computational time compared to its centralized counterparts, while adding a small fraction of latency compared to the non-privacy preserving distributed schemes. In the future, the proposed method will be extended and applied to practical applications which require learning with strict security and a high accuracy.

Acknowledgments

This work is supported in part by the National Natural Science Funds for Distinguished Young Scholar under Grant 62325307, in part by the National Nature Science Foundation of China under Grants 62072315, 62073225, 62176164, and 62203134, in part by the Shenzhen Science and Technology Program under grants KCXFZ20230731094001003, 20220809141216003, JCYJ20210324093808021, and JCYJ20220531102817040, in part by the Scientific Instrument Developing Project of Shenzhen University under Grant 2023YQ019, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2022A1515012326, in part by the Guangdong "Pearl River Talent Recruitment Program" under Grant 2019ZT08X603, in part by the Guangdong "Pearl River Talent Plan" under Grant 2019JC01X235, and Shenzhen Talents Special Project - Guangdong Provincial Innovation and Entrepreneurship Team Supporting Project (Grant 2021344612).

References

Brakerski, Z. 2012. Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. In *Advances in Cryptology (CRYPTO)*, 868–886.

Chen, F.; Li, P.; and Miyazaki, T. 2021. In-Network Aggregation for Privacy-Preserving Federated Learning. In *2021 International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*, 49–56.

Chen, H.; Dai, W.; Kim, M.; and Song, Y. 2019. Efficient Multi-Key Homomorphic Encryption with Packed Ciphertexts with Application to Oblivious Neural Network Inference. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, (CCS), 395–412. New York.

Chen, J.; Cao, N.; Low, K. H.; Ouyang, R.; Tan, C. K.-Y.; and Jaillet, P. 2013. Parallel Gaussian process regression with low-rank covariance matrix approximations. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, (UAI), 152–161.

Chen, J.; Low, K. H.; Tan, C. K.-Y.; Oran, A.; Jaillet, P.; Dolan, J.; and Sukhatme, G. 2012. Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, (UAI), 163–173.

Cheon, J. H.; Kim, A.; Kim, M.; and Song, Y. 2017. Homomorphic Encryption for Arithmetic of Approximate Numbers. In *Advances in Cryptology (ASIACRYPT)*, 409–437.

Csató, L.; and Opper, M. 2002. Sparse on-line gaussian processes. *Neural Computation*, 14(3): 641–668.

Dwork, C. 2008. Differential Privacy: A Survey of Results. In *Theory and Applications of Models of Computation*, Lecture Notes in Computer Science, 1–19.

Fenner, P.; and Pyzer-Knapp, E. 2020. Privacy-Preserving Gaussian Process Regression – A Modular Approach to the Application of Homomorphic Encryption. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34: 3866–3873.

Gentry, C. 2009. *A fully homomorphic encryption scheme*. Phd thesis, Stanford University, Stanford, USA.

Horvitz, E.; and Mulligan, D. 2015. Data, privacy, and the greater good. *Science*, 349(6245): 253–255.

Lederer, A.; Conejo, A. J. O.; Maier, K. A.; Xiao, W.; Umlauft, J.; and Hirche, S. 2021. Gaussian Process-Based Real-Time Learning for Safety Critical Applications. In *Proceedings of the 38th International Conference on Machine Learning, Virtual Event (ICML)*, 6055–6064.

Liu, Z.; Guo, J.; Yang, W.; Fan, J.; Lam, K.-Y.; and Zhao, J. 2022. Privacy-Preserving Aggregation in Federated Learning: A Survey. *IEEE Transactions on Big Data*, 1–20.

Ma, J.; Naas, S.-A.; Sigg, S.; and Lyu, X. 2022. Privacy-preserving federated learning based on multi-key homomorphic encryption. *International Journal of Intelligent Systems*, 37(9): 5880–5901.

Mandal, K.; and Gong, G. 2019. 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*, (CCSW), 57–68. New York, NY, USA.

Melis, L.; Song, C.; De Cristofaro, E.; and Shmatikov, V. 2019. Exploiting Unintended Feature Leakage in Collaborative Learning. In *2019 IEEE symposium on security and privacy (SP)*, 691–706.

Mohassel, P.; and Rindal, P. 2018. ABY3: A Mixed Protocol Framework for Machine Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, (CCS), 35–52. New York, NY, USA.

Naseri, M.; Hayes, J.; and De Cristofaro, E. 2022. Local and Central Differential Privacy for Robustness and Privacy in Federated Learning. ArXiv:2009.03561 [cs].

Nugteren, C.; and Codreanu, V. 2015. CLTune: A Generic Auto-Tuner for OpenCL Kernels. In *2015 IEEE 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip*, 195–202. Turin, Italy.

Phong, L. T.; Aono, Y.; Hayashi, T.; Wang, L.; and Moriai, S. 2018. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. *IEEE Transactions on Information Forensics and Security*, 13(5): 1333–1345.

Rasmussen, C. E. 2003. Gaussian processes in machine learning. In *Summer school on machine learning*, 63–71. Springer.

Salam, A.; and Hibaoui, A. E. 2018. Comparison of Machine Learning Algorithms for the Power Consumption Prediction : - Case Study of Tetouan city –. In *2018 6th International Renewable and Sustainable Energy Conference (IRSEC)*, 1–5. Rabat, Morroco.

Seeger, M. W.; Williams, C. K. I.; and Lawrence, N. D. 2003. Fast Forward Selection to Speed Up Sparse Gaussian

Process Regression. In *International Workshop on Artificial Intelligence and Statistics*, 254–261. Key West, Florida, USA.

Snelson, E.; and Ghahramani, Z. 2005. Sparse Gaussian Processes using Pseudo-inputs. In *Advances in Neural Information Processing Systems*, volume 18.

Tang, F.; Wu, W.; Liu, J.; Wang, H.; and Xian, M. 2019. Privacy-Preserving Distributed Deep Learning via Homomorphic Re-Encryption. *Electronics*, 8(4): 411.

Vijayakumar, S.; and Schaal, S. 2000. Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space. In *Proceedings of the seventeenth international conference on machine learning (ICML 2000)*, volume 1, 288–293. Stanford, CA, USA.

Xu, R.; Baracaldo, N.; Zhou, Y.; Anwar, A.; Joshi, J.; and Ludwig, H. 2021. FedV: Privacy-Preserving Federated Learning over Vertically Partitioned Data. In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, (AISec), 181–192. New York, NY, USA.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2): 12:1–12:19.

Yao, A. C. 1982. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs)*, 160–164.

Yuan, J.; and Yu, S. 2014. Privacy Preserving Back-Propagation Neural Network Learning Made Practical with Cloud Computing. *IEEE Transactions on Parallel and Distributed Systems*, 25(1): 212–221.

Zhang, X.; Fu, A.; Wang, H.; Zhou, C.; and Chen, Z. 2020. A Privacy-Preserving and Verifiable Federated Learning Scheme. In *2020 IEEE International Conference on Communications (ICC 2020)*, 1–6.

Zhou, C.; Fu, A.; Yu, S.; Yang, W.; Wang, H.; and Zhang, Y. 2020. Privacy-Preserving Federated Learning in Fog Computing. *IEEE Internet of Things Journal*, 7(11): 10782–10793.

Zhu, H.; Wang, R.; Jin, Y.; and Liang, K. 2021. PIVODL: Privacy-preserving vertical federated learning over distributed labels. *IEEE Transactions on Artificial Intelligence*, 1–1.