Sigg, Stephan; Nguyen, Le Ngu; Ma, Jing

Camouflage Learning

Published in:
2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events, PerCom Workshops 2021

DOI:
10.1109/PerComWorkshops51409.2021.9431111

Published: 25/05/2021

Please cite the original version:
Camouflage Learning

Stephan Sigg, Le Ngu Nguyen, Jing Ma
Department of Communications and Networking
Aalto University, Finland
{stephan.sigg, le.ngu.nguyen, jing.ma}@aalto.fi

Abstract—Federated learning has been proposed as a concept for distributed machine learning which enforces privacy by avoiding sharing private data with a coordinator or distributed nodes. However, information on local data might be leaked through the model updates. We propose Camouflage learning, a machine learning scheme that distributes both the data and the model. Neither the distributed devices nor the coordinator is at any point in time in possession of the complete model. Furthermore, data and model are obfuscated during distributed model inference and distributed model training. Camouflage learning can be implemented with various Machine learning schemes.

Index Terms—Internet of Things, Distributed machine learning, Privacy, Multi-key homomorphic encryption

I. INTRODUCTION

IoT devices are characterized by their connectivity, sensing and processing capabilities. The vast number of IoT devices holds a tremendous potential for personalized services and ambient intelligence. Distributed sensing from on-body and environmental devices enables personalized services, behavior perception, and scene understanding through artificial intelligence. In particular, distributed machine learning, such as federated learning fosters such scenario of highly distributed, computationally constrained devices, and also addresses a frequently raised issue in such IoT environments: how to protect data privacy in distributed sensing and computation.

In federated learning, each participant conducts model training locally, and then sends the model updates to the coordinator to optimize a global model. Hence, privacy of data is provided despite the distributed nature of the scheme. However, it was shown that information on the data leaks from the local model updates. The main weakness is that the model updates can be observed and analyzed in order to infer private information on the local data. Several authors have suggested modifications of the classic federated learning scheme to provide data privacy.

We propose the total distribution of the model to address information leakage issue from model updates. In particular, in Camouflage learning, weights are kept and updated locally with the data. When weights and data are shared with the coordinator for model inference, their absolute values are obfuscated by sharing the weighted sum instead of the individual values. In addition, we suggest the use of multi-key homomorphic encryption to prevent eavesdropping on the shared information. At the coordinator, encrypted weighted sums are aggregated and finally decrypted for model inference by the coordinator.

II. RELATED WORK

Federated learning has been proposed to build machine-learning models based on distributed datasets [4]. In particular, assume $n$ data owners $\{1, \ldots, n\}$, with respective data $\{x_1, \ldots, x_n\}$, that collaboratively train a machine learning model $M$, without exposing the data $x_i$ to others. Early federated learning models have shared model updates for an optimization algorithm such as stochastic gradient descent [5], [6], which may leak information on the data [7]–[10].

In horizontal federated learning, datasets $x_i$ share the same feature space but differ in sample space [11]. In contrast, in vertical federated learning, features from respective data samples are spread across devices. This case resembles IoT applications, where multiple devices observe features from common stimuli (cf. figure 1). Privacy preserving machine learning algorithms for vertical federated learning comprise, for instance, cooperative statistical analysis [14], association rule mining [15], secure linear regression [16], classification [17], and gradient descent [18]–[20].

Secure multi-party computation (SMC) [21] assumes that multiple parties provide a security proof to guarantee complete zero knowledge, at the expense of demanding computation. In these related works, private data $x_i$ need to be shared secretly among server and parties [22]. Furthermore, differential privacy [23], diversification [24] or k-anonymity [25], such as [26], [27] add noise or use generalization models to obscure the data, so that an adversary is unable to derive individual information. The authors of [28] have proposed differential privacy for federating learning.
Finally, homomorphic encryption [2], [3] has been proposed to protect the distributed data \(x_i\) [29], [30]. The data \(x_i\) is combined after being encrypted and shared, thereby directly addressing the issue of information leakage. Due to its efficiency, additive homomorphic encryption is often used, which, however, requires polynomial approximations on nonlinear functions in machine learning algorithms, and results in accuracy and privacy tradeoffs [31], [32].

III. HOMOMORPHIC ENCRYPTION

An encryption scheme \(E\) is called homomorphic over a message space \(M\) with respect to an operation \(\circ\) if

\[
E[m_1] \circ E[m_2] = E[m_1 \cdot m_2], \quad \forall m_1, m_2 \in M.
\]

(1)

If the operator \(\circ\) can be both multiplication AND addition, we call this scheme a fully-homomorphic scheme. If it supports only one (either multiplication OR addition), it is partially homomorphic [3]. Popular additive homomorphic encryption schemes are, for instance, by Goldwasser and Micali [33], Paillier [34], Damgard and Jurik [35], or Kawachi et al. [36]. The first fully homomorphic encryption scheme [37] has seen a lot of follow-up improvements to address its limitations with regard to computational load. Homomorphic encryption can be symmetric (same key for encryption and decryption), as well as asymmetric (different keys) [38].

For our work, we propose to use the fully homomorphic encryption system introduced by López-Alt et al. [39], which features the multi-key property that homomorphic operations are possible on messages \(E_{k_i}[m_i]\) and \(E_{k_j}[m_j]\) encrypted by different keys \(k_i\) and \(k_j\) with \(i \neq j\). Further properties that are useful in our construction (cf. section IV) are:

1. the scheme is asymmetric and (2) it can be constructed in such a way that decryption is only allowed after all messages \(m_i, i \in \{1, \ldots, n\}\) have been combined as \(\sum_{i=1}^{n} E_{k_i}[m_i]\).

In particular, polynomials \(f_i, g_i, s_i, e_i\) with \(i \in \{1, \ldots, n\}\) are sampled from a discrete Gaussian distribution \(\chi\) and invertible functions \(f_i = 2f_i^1 + 1\) are defined. Secret keys for IoT devices \(d_i\) are chosen as \(k_i^{sec} = f_i\) with the local key as \(k_i^{loc} = 2g_i f_i^{-1}\). A message \(m_i\) is encrypted as

\[
E_{k_i}[m_i] = k_i^{loc} s_i + 2e_i + m_i.
\]

(2)

Likewise, the message is decrypted with

\[
f_i E_{k_i}[m_i] \equiv f_i (k_i^{loc} s_i + 2e_i + m_i) \pmod{2}
\]

\[
= f_i (2g_i f_i^{-1} s_i + 2e_i + m_i) \pmod{2}
\]

\[
= 2(g_i s_i + e_i f_i) + f_i m_i \pmod{2}
\]

\[
= f_i m_i \pmod{2}
\]

\[
= 2f_i m_i + m_i \pmod{2}
\]

\[
= m_i
\]

(3)

This fully homomorphic encryption scheme further has the property that mathematical operations can be conducted on encrypted messages \(E_{k_i}[m_i]\) and \(E_{k_j}[m_j]\) with \(i \neq j\) and the decryption key is simply the product of the respective secret keys \(k_i^{sec}, k_j^{sec}\):

\[
k_i^{sec} \cdot k_j^{sec} (E_{k_i}[m_i] + E_{k_j}[m_j]) = m_i + m_j,
\]

(4)

and hence

\[
\prod_{i=1}^{n} k_i^{sec} \left( \sum_{i=1}^{n} E_{k_i}[m_i] \right) = \sum_{i=1}^{n} m_i,
\]

(5)

so that a coordinator in possession of \(\prod_{i=1}^{n} k_i^{sec}\) is able to decrypt the sum over all homomorphically encrypted messages from distributed devices \(d_i\), if and only if all \(E_{k_i}[m_i]\) have been included in the sum [39].

IV. CAMOUFLAGE LEARNING

Assume a scenario with a single coordinator and \(n\) distributed devices \(d_1, \ldots, d_n\) which command shares of the data \(x_1, \ldots, x_n\) as well as respective weights \(w_1, \ldots, w_n\). Each device possesses a local key \(k_i^{loc}\) for an additive homomorphic encryption system \(E^+(\cdot)\). The coordinator \(c\) is in possession of a key \(k_c = \prod_{i=1}^{n} k_i^{sec}\). In addition, the IoT devices \(d_i\) further share a value \(\lambda\), which controls the convergence speed of a gradient descent mechanism.

Note that, for this initialization, the respective keys \(k_i^{loc}\), as well as \(k_c\) can either be distributed to the participating devices by a centralized entity (introducing a possible single point of failure for an adversarial attack), or be computed independently. In the latter case, the \(k_i^{loc}\) can be computed independently by the remote IoT devices \(d_i\). Only knowledge on the discrete Gaussian distribution \(\chi\) need to be shared across devices. For the generation of the key \(k_c\), the respective \(k_i^{sec}\), which are in the possession of the \(d_i\), need to be combined without disclosing information on the keys to other devices in the process. We suggest to use the secure product protocol from [41] for this process. The idea of the protocol is that each \(k_i^{sec}\) is split into random terms \(k_{ij}^{sec}\) with

\[
\prod_{j=1}^{m} k_{ij}^{sec} = k_i^{sec}
\]

(6)

From the \(m\) terms \(k_{ij}^{sec}\) at each node, \(m - 1\) splits are then randomly distributed via 2-party encrypted channels across IoT devices (i.e. not disclosing one of the \(m\) terms), multiplied together at each \(d_i\) locally, and shared with the coordinator \(c\), which will in turn compute

\[
k_c = \prod_{i=1}^{n} k_i^{sec} = \prod_{i=1}^{n} \prod_{j=1}^{m} k_{ij}^{sec}
\]

(7)

as product of all received values. We refer the interested reader to [41] for further details on securely computing a product from \(n\) terms \(k_i^{sec}\).

A. Model inference

For model inference in Camouflage learning, the coordinator requests from the distributed IoT devices \(d_1, \ldots, d_n\) their weighted features values \(w_1 x_1, \ldots, w_n x_n\).
On receiving this request, devices \(d_1, \ldots, d_n\) send \(E_{k_e}^+[w_ix_i]\), their weighted feature values \(w_ix_i\), encrypted with additive homomorphic encryption, to the coordinator \(c\) (cf. figure 2).

Since the keys \(k_{i_{sec}}^e\) are only known to \(d_i\), the coordinator is not able to decrypt the received \(E_{k_e}^+[w_ix_i]\). However, because the weighted feature values are encrypted with additive homomorphic encryption, the coordinator can compute

\[
E_{k_e}^+[W^TX] = \sum_{i=1}^{n} E_{k_e}^+[w_ix_i].
\]

For \(E_{k_e}^+[W^TX]\), the coordinator can use \(k_c\) for decryption and hence obtain

\[
k_c(E_{k_e}^+[W^TX]) = W^TX = \sum_{i=1}^{n} w_ix_i.
\]

Note that the coordinator is not able to obtain information about the \(x_i\) or \(w_i\) since these values are obfuscated in two ways: the devices \(d_i\) only share the weighted feature values \(w_ix_i\) and not the weights \(w_i\) or the feature values \(x_i\) separately. Furthermore, since these values are encrypted as \(E_{k_e}^+[w_ix_i]\), the coordinator can only obtain the weighted sum over all feature values, \(\sum_{i=1}^{n} w_ix_i\), which further reduces the likelihood of \(c\) obtaining any information on \(x_i\) or \(w_i\).

Without loss of generality, we assume that logistic regression is implemented via the distributed Camouflage learning. In section VI-B, the use of alternative machine learning schemes is discussed.

For logistic regression, to infer a classification from the model, the coordinator computes a function in the form

\[
h(X) = \frac{1}{1 + e^{-W^TX}} + \delta
\]

(10)

Note that, knowing \(W^TX\), as well as \(\delta\), the coordinator, even though it is ignorant of \(X\), is still able to compute \(h(X)\).

Summarizing, Camouflage learning distributes the model over all IoT devices and the coordinator, where each device \(d_i\) is aware of its own feature values \(x_i\) and corresponding weights \(w_i\) only, while only the coordinator is aware of \(\delta\).

No single node, and not even the coordinator is aware of the complete model. Hence, an adversary, regardless of whether it is an external eavesdropper, a compromised device \(d_j\) or the coordinator \(c\), is not capable of deriving information on \(w_i\) or \(x_i\) with \(i \in \{1, \ldots, n\}\), \(i \neq j\).

B. Model training

The weights \(w_i\) are distributed on the IoT devices \(d_i, i \in \{1, \ldots, n\}\). Hence, for model training, the local weights \(w_i\) have to be updated by the respective devices \(d_i\). For this, an iterative protocol is implemented between the coordinator \(c\) and the IoT devices \(d_1, \ldots, d_n\) (cf. figure 3).

In particular, for a class \(y\), the coordinator requests the weighted feature values \(w_ix_i\) from the distributed IoT devices \(d_i\). Each device \(d_i\), in turn, sends \(E_{k_i}^+[w_ix_i]\) to the coordinator, which aggregates these values (equation (8)), decrypts with \(k_c\), and computes the loss

\[
L(W^TX) = -y \log (h(X)) + (1 - y) \log (1 - h(X)).
\]

(11)

The loss \(L(W^TX)\) is then shared with all IoT devices \(d_1, \ldots, d_n\), which, in turn, update their local weights as

\[
u_i^{\text{new}} = v_i + \lambda \frac{\partial}{\partial w_i} L(W^TX).
\]

(12)

This process is iterated until convergence.

Note that an adversary, even if it is a compromised IoT device \(d_j\) or even a compromised coordinator \(c\) is not able to learn any weights \(w_i\) or feature values \(x_i\) of any other devices \(d_i, i \in \{1, \ldots, n\}, i \neq j\), because neither the weights nor the features are transmitted individually at any point in time but instead as weighted feature values \(w_ix_i\). On top of this, they are encrypted during transmission as \(E_{k_i}^+[w_ix_i]\) and not decrypted before being combined to their weighted sum \(\sum_{i=1}^{n} E_{k_i}^+[w_ix_i]\).

V. EVALUATION

We simulated Camouflage learning on an IoT application: occupancy monitoring. Candanedo and Feldheim [42] developed a system of light, temperature, humidity, and CO₂...
sensors to collect occupancy data in an office. Each sample was the average value captured by these sensors over 60 seconds. The ground-truth was annotated using videos. They provided two datasets: six-day data for training and nine-day data for testing. We pre-processed the data by scaling samples according to their minimum and maximum sensing values. Data was then gradually sent to the coordinator; hence, the resource consumption was accumulated. Figure 4 visualizes the encryption time and the ciphertext size accumulated during the training process. Apparently, the model performance improved with increasing the ciphertext size produced a ciphertext of 100KB. Since we gradually sent one value, a device required 0.296 seconds and (a) Total ciphertext size for each IoT device (b) Total encryption time on each IoT device

Fig. 4: Resource consumption in each IoT device when sending encrypted data to the coordinator

VI. DISCUSSION

In this section, we briefly discuss possible security threats for Camouflage learning, other machine learning schemes that are compatible with Camouflage learning, as well as an alternative implementation for non star-type network topologies.

A. Threat analysis

Figure 5 depicts possible attack surfaces and attack vectors for Camouflage learning. In particular, the protocol may be attacked either on the level of the coordinator, on the level of the communication between devices \(d_1, \ldots, d_n\) or on the level of the IoT devices.

A possible risk for the devices (coordinator or IoT devices) is that an adversary might gain remote or physical access to them (e.g. compromised or malicious device) \((1)\) and \((9)\) in figure 5). However, since the model is totally distributed in Camouflage learning, the damage in such case is limited. The adversary would gain access only to the local data \(x_i, w_i, f_i, g_i, s_i, e_i, \lambda, X\) (or to \(\delta, k_c\) and \(W^TX\)). An adversary able to compromise the coordinator might further launch a brute force attack on the key \(k_c\) to extract \(\prod_{i=1}^{n} E_{k_c}[w_i x_i]\) \((2)\) in figure 5). However, it is very unlikely, that the \(n\) terms of the product would be accidentally guessed correctly. Other possible attacks regard the extraction of the \(w_i\) and \(x_i\) from \(\sum_{i=1}^{n} w_i x_i\) \((3)\) in figure 5), which is equally unlikely, or an attack on the model \((4)\) in figure 5), which again is unlikely to succeed, because the individual contributions \(w_i\) and \(x_i\) on the model updates are obscured with the contributions from all other devices as \(\sum_{i=1}^{n} w_i x_i\).

Furthermore, a man-in-the-middle attack \((5)\) in figure 5) could be launched on the communication between any two parties in the protocol. While this could be successful, the adversary could not learn anything as the devices do not leak any private information across each other. Furthermore, an adversary might attack the multi-key agreement protocol among the devices \(d_1, \ldots, d_n\) to obtain information about the \(k_i\), or she might attack the homomorphically encrypted data \(E_{k_i}[w_i x_i]\) \((6)\) and \((7)\) in figure 5). This attack would assume a weakness in the protocols described in [41] or [39]. Also, the adversary could launch a denial of service attack against the communication between the IoT devices or she might replay previous transmitted data or try to insert own data \((8)\) in figure 5). While the latter (inserting own data) attempt can not be successful in absence of the correct encryption key \(k_i\), the denial of service or reply attacks could be successful. However, the adversary would then at most be able to disturb the model training or inference, but not learn anything about the model or the data of any device \(d_i\).

Finally, an adversary with physical or remote access to one or more of the IoT devices \(d_1, \ldots, d_n\) might obtain data from the device itself or also insert new data \((10)\) in figure 5). The impact would, however, be limited as the learned information or the inserted data regards a single device only. In addition, an adversary might exploit a possible bias in the key generation on devices \(d_1, \ldots, d_n\). This attack would assume a weakness in the protocols described in [39]. Finally, a weakness in the pseudo-random number generator utilized by the devices \(d_1, \ldots, d_n\) might be exploited by an adversary to gain

Fig. 5: Attack vectors and attack surfaces of Camouflage Learning

<table>
<thead>
<tr>
<th>Attack Vectors</th>
<th>Attack Surfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Spoofing / compromise</td>
<td>Coordinator</td>
</tr>
<tr>
<td>(2) Brute force</td>
<td>multi-key agreement</td>
</tr>
<tr>
<td>(3) Biased feature processing</td>
<td>weighted features</td>
</tr>
<tr>
<td>(4) model analysis / leak</td>
<td>model</td>
</tr>
<tr>
<td>(5) Man-in-the-Middle attack</td>
<td>Communication</td>
</tr>
<tr>
<td>(6) Weak key agreement</td>
<td>multi-key agreement</td>
</tr>
<tr>
<td>(7) Weak Encryption scheme</td>
<td>encrypted data</td>
</tr>
<tr>
<td>(8) Denial of service / replay / insertion</td>
<td>protocol</td>
</tr>
<tr>
<td>(9) Spoofing / compromise</td>
<td>IoT devices</td>
</tr>
<tr>
<td>(10) Replay / bypass data acquisition</td>
<td>sensors</td>
</tr>
<tr>
<td>(11) Bias in key generation</td>
<td>key generation</td>
</tr>
<tr>
<td>(12) Biased random number generation</td>
<td>PRNG</td>
</tr>
</tbody>
</table>
advantage in decoding any of the encrypted data $E_{k_i}[w_ix_i]$. This is indeed a serious threat as IoT devices might run on outdated/non-updated device firmware. We remark that such weakness would essentially compromise any secure algorithms executed on the IoT devices and not only Camouflage learning.

B. Other Machine learning schemes

We have discussed the implementation of Camouflage learning using the example of logistic regression. However, Camouflage learning can be implemented also for other machine learning schemes, provided that they expect a sum over the weighted feature values $\sum_{i=1}^{n} x_iw_i$.

For instance, other regression-type problems (for instance, linear-, polynomial-, multivariable-, multivariate-, logistic regression) are straightforward to implement as Camouflage learning scheme.

Furthermore, powerful classifiers, that can be implemented as Camouflage learning scheme across distributed IoT devices, are support vector machines. In particular, for a binary classification problem with $y \in \{0, 1\}$ and an appropriate cost function $\text{cost}_y$, the loss of a support vector machine is defined as

$$L_{\text{SVM}}(W^TX) = -y \cdot \text{cost}_{y=1}(W^TX) - (1 - y) \cdot \text{cost}_{y=0}(W^TX).$$

(13)

For model inference, similar as for regression-type classifiers, knowledge on the sum over the weighted feature values $W^TX$ is sufficient.

Also, artificial neural networks can be partially implemented according to the Camouflage learning scheme. In particular, a neural network with $M$ layers including the input layer, dimension $D_i$, and an activation function $f(i)$ on layer $i$, can generally be written in the form

$$g_{\text{ANN}}(X, W) = f_{\text{act}}^{(M)}(z^{(M-1)})$$

(14)

with

$$z^{(p)}_j = \sum_{r=1}^{D_{p-1}} w_{rj}^{(p-1)} f_{\text{act}}^{(p-1)}(z^{(p-1)}_r) + w_{0j}^{(p-1)},$$

(15)

$$z^{(2)}_j = \sum_{i=1}^{D_1} w_{ij}^{(1)} x_i + w_{0j}^{(1)}.$$  

(16)

Hence, the input of the weighted features in the first layer (equation (16)) can again be distributed across IoT devices $d_1, \ldots, d_n$, and transmitted encrypted to the coordinator. For Camouflage learning, the weight updates have to follow a gradient descent model for artificial neural networks.

C. Alternative implementations

The described implementation of Camouflage learning assumes the existence of a dedicated central coordinator. However, the implementation of such star-structure might not be feasible in some IoT scenarios. Furthermore, the multi-key homomorphic encryption requires some computational and communication overhead during the bootstrap phase (equation (7)). We invite fellow researchers to come up with alternative implementations of Camouflage learning, that foster distinct IoT scenarios and device topologies.

A possible alternative implementation is depicted in figure 6. Devices are assigned a unique order and device $d_j$ will, on deriving $\sum_{i=1}^{j} E_{k_i}[w_ix_i]$, forward

$$\sum_{i=1}^{j} E_{k_i}[w_ix_i] = \left( \sum_{i=1}^{j-1} E_{k_i}[w_ix_i] \right) + E_{k_j}[w_jx_j]$$

(17)

to device $d_{j+1}$. Device $d_n$ sends $\sum_{i=1}^{n} E_{k_i}[w_ix_i]$ to the coordinator.

This implementation achieves the same computation as the Camouflage model described above but does not expect a central coordinator and a star-topology across IoT devices. The communication and processing load of the central coordinator is significantly lower (by factor $n$) since it receives $\sum_{i=1}^{n} E_{k_i}[w_ix_i]$ from device $d_n$, instead of receiving $E_{k_i}[w_ix_i]$ from $d_1, \ldots, d_n$. However, note that the communication cost for individual IoT devices $d_i$ increases by approximately factor 2 and the chaining of the transmissions requires further administrative overhead. The alternative model achieves a fair share of the processing and communication load and is in particularly well suited when the coordinator is a resource constrained IoT device.

VII. CONCLUSION

We proposed Camouflage learning, a distributed machine learning scheme that obfuscates model and data and thereby achieves privacy for participating IoT devices. In contrast to federated learning, it does not leak information on the local data. Furthermore, Camouflage learning maximally distributes the machine learning model, so that it is at no point in time completely known to any party, including the coordinator. To implement Camouflage learning, we utilized multi-key additive homomorphic encryption. Individual devices share their weighted features encrypted and the coordinator combines the encrypted components via addition. For model inference, the coordinator utilized a specific decryption key to decrypt the weighted sum of the feature values and computes the model inference. Model training is achieved via a distributed gradient descent mechanism.

An alternative implementation for Camouflage learning, utilizing chained inference transmission has also been discussed.
This alternative model does not require a central coordinator, thereby significantly reducing the communication load and energy consumption of the coordinator, while doubling the communication load for all other IoT devices.

ACKNOWLEDGMENT
We would like to acknowledge partial funding by the Academy of Finland Project ABACUS (ICT 2023).

REFERENCES