
On Private Supervised Distributed Learning: Weakly Labeled and without Entity Resolution

Stephen Hardy[†] Wilko Henecka[†] Richard Nock^{†,‡}

[†]Data61; [‡]The Australian National University & the University of Sydney, Australia
first_name.last_name@data61.csiro.au

Abstract

We describe a system with strong privacy guarantees that is able to learn (supervised) linear classifiers in the challenging setting where data is *distributed*, entity matching/resolution is *not* possible and *not all parties have labels*. The privacy guarantees are due to the use of Rademacher observations (rados) for learning, where these rados are calculated using only the functionality provided by *partially* homomorphic encryption, which obscures the source data and provides realistic learning times. Differentially private Rados can also be computed, leading to a distributed machine learning algorithm guaranteeing the protection of both the data sources (data remains private to the contributors) and its aftermath. An illustration of the performances of the models is provided on two (synthetic+UCI) datasets.

1 Introduction

Machine Learning (ML) is more and more frequently subsumed by the more complex goal of learning in a private setting [Enserink and Chin, 2015, Goroff, 2015]. More generally, the notion of *data security* is a broad field concerned with fairness, accountability, correctness, secrecy and availability issues regarding data [Canetti, 2006] — all these apply to ML to various extents depending on the application domain, and in all cases, there is no tradeoff nor compromise with those requirements: they are mandatory and thus adds constraints that essentially cannot be relaxed. Potential losses for ML are from the computational [Dowlin et al., 2016] and accuracy standpoints [Goroff, 2015].

There are three essential settings to protect information or its aftermath [Dwork and Roth, 2014, Goldreich, 1987]. Some naturally fit to learning from distributed data. *Secure multiparty computation* (SMC) is one: $p > 1$ peers compute the input of a given function of all parties data, while keeping their input secret from all other peers. Secrecy is guaranteed up to varying amounts of adversarial behaviour, which can be collusion between parties or various degrees of honesty in the behaviour of parties. (*Homomorphic*) *public-key encryption* (PKE) theoretically completely obfuscates data except from the owner (of the private key). Homomorphic properties allow to perform certain operations on the data in the clear ("plaintexts") via equivalent operations on cyphertexts. For example, in Paillier cryptosystem [Paillier, 1999], the multiplication of two cyphertexts gives the encryption of the sum of the two plaintexts. In general, secrecy relies on assumptions on the computational hardness of specific decision problems, such as the decisional composite residuosity assumption for Paillier encryption [Paillier, 1999]. Finally, *Differential privacy* (DP) modifies data with noise and provides protection in a plausible deniability model: one can control how his/her presence in some data is going to affect the outcome of some mechanisms [Dwork and Roth, 2014].

There are two key facts about these notions of privacy: they protect different things about data and little to none are in fact 100% bullet proof [Kifer and Machanavajjhala, 2011] (at least up to some assumptions). This suggests that *composing* privacy guarantees may be interesting to improve protection, but this potentially comes with a composition of the associated constraints for learning, and therefore may reduce its efficiency drastically. For example, when dealing with PKE, constraints

Method	Computes	Protects	Breach ?	Ref.
SMC	Distributed function	Information sources	Collusion, dishonesty	[Goldreich, 1987]
PKE	Information sharing	Information sources	Breaking assumptions	[Goldreich, 1987]
DP	Information sharing	Information's <i>aftermath</i>	Side information	[Kifer and Machanavajjhala, 2011]

Table 1: Comparison of SMC (secure multiparty computation), PKE (public key encryption) and DP (differential privacy) from different standpoints. "Protection" summarizes the kind of protection they provide. "Breach ?" summarizes the known breaches of privacy (with references, see text).

are such that the state of the art focusing on reasonably complex models (beyond linear separators) actually just performs the *classification* step, not even learning [Bost et al., 2014, Dowlin et al., 2016].

In this paper, we provide the overview of a system that focuses on learning linear separators, in a setting with the following key properties: (i) data is distributed among $p \geq 1$ peers (or parties), parties *share* some categorical features (*e.g.* gender, postcode, etc.). The classifier is learned on the union of *all* peers' features, *but* we do not need to perform *entity matching* (the task which consists in finding who is who among datasets); finally, the task is *weakly labeled* as not all available data may be labeled — in our case, only a subset of the parties may have labels. (ii) local label learning (LLP) is achieved under Paillier cryptosystem's *encrypted domain* — subsequent operations, up to learning the classifier itself, can also be performed in the encrypted domain. Our protection level is therefore primarily relevant to PKE (Table 1), but the technology we use, that builds upon recent results on loss factorization and optimization [Nock, 2016, Nock et al., 2015, Patrini et al., 2014, 2016a,b], allows to include DP / SMC related protections as well (*i.e.* protections that rely on computational or statistical guarantees related to information sharing or the distributed computation itself [Nock et al., 2015]).

The rest of this paper is organized as follows: Sections 2 and 3 present key definitions and sketch our system. Section 4 provides experiments. A last Section (5) concludes.

2 Definitions

Privacy notions: Table 1 summarizes the three different types of protection we are interested in, their purpose, the protection they give and the eventual forms of (known) breaches. We refer to Goldreich [1987] for more details on SMC and PKE, and Kifer and Machanavajjhala [2014] (and references therein) for DP. Our system's protection essentially covers PKE and DP.

Learning setting: our setting is binary classification with linear models θ : we want to learn a predictor or classifier for label (or class) $y \in \mathcal{Y} \doteq \{-1, +1\}$, where the classifier is a function from an observation (or feature) space, \mathcal{X} , to \mathcal{Y} . As in supervised learning, the input of learning are examples, that is, observation-label pairs. However, in our distributed setting, we do not have access to a training sample $\mathcal{S} \doteq \{(\mathbf{x}_i, y_i), i \in [m]\}$ ($[n] \doteq \{1, 2, \dots, n\}$ for $n \in \mathbb{N}_*$). Rather, we have p (sub)samples (one for each party), \mathcal{S}^j of size m_j , $j \in [p]$ for some $p \geq 1$. Each one is defined on its own feature space $\mathcal{X}^j \doteq \times_{k \in \mathcal{F}_j} X_k$, where $\mathcal{F}_j \subseteq [d], \forall j$ (d is the total number of distinct features among peers). We assume that a subset of features is shared: $\exists \mathcal{F}^* \subset [d]$ such that $\mathcal{F}^* \subseteq \mathcal{F}_j, \forall j \in [p]$. This setting is realistic: in many scenarios, all peers (including *e.g.* insurance companies, banks, telecommunication companies, government agencies) would share at least few features in their data, such as birthdate, postcode, gender of their customers. Figure 1 (left) presents the overall distributed learning setting. We also assume that even when not all peers have labels, at least *one* peer has its data fully labeled. It is challenging but very relevant not to assume that all parties have labels, *e.g.* for a specific credit scoring application in which a bank leverages upon the unsupervised data of an insurance company. The learning setting (weakly labeled, no entity matching) blends those of [Patrini et al., 2014, 2016b].

Homomorphic cryptosystem: we use a PKE cryptosystem with (partial) *homomorphic* properties: Paillier encryption [Paillier, 1999]. Given number n and associated encryption $\llbracket n \rrbracket$, the homomorphic property states that the encryption of some function f of n, n' can be computed as a function g of their encryption: $\llbracket f(m, m') \rrbracket = g(\llbracket m \rrbracket, \llbracket m' \rrbracket)$. Paillier is additively homomorphic (f =arithmetic addition). It also allows the multiplication of an encrypted number by a plaintext number.

Compute setting: our machine learning system is built on top of a REST-ful distributed computing API implemented in Java, with cryptographic primitives supported natively within the system,

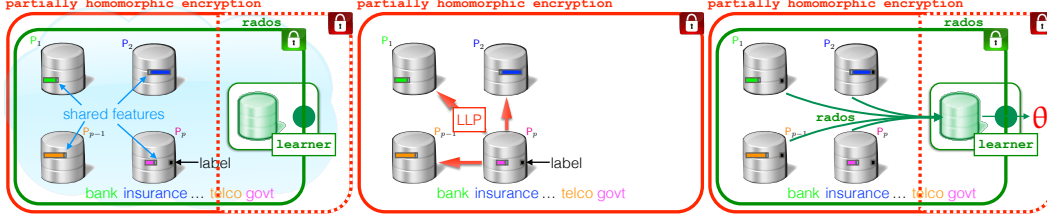



Figure 1: (Left) Our setting: $p > 1$ parties, one (or few, not necessarily all) have labels, each has observation features *distinct* from the others, all share some common features, and *no* entity matching can be performed between parties. Our system proposes an end-to-end solution for supervised learning of θ in a distributed and private setting (with respect to encryption *and* additional privacy guarantees provided by rados, see text). θ classifies using *all* peers’ features (*i.e.* maps  to classes). *Center+right*: learning is performed in two steps. First (center), all parties compute their own labels using (private) learning from label proportions. Second (right) each party crafts its own share of basic block rados sent to the learner, which assembles rados and learns θ .

including Paillier encryption¹. The network transport layer is HTTPS, and there is an additional authorisation layer which specifies which parties can compute over which data with which algorithms. Currently the system supports a variety of machine learning tasks, including the linear supervised learning approaches described here, as well as a variety of unsupervised learning tasks over both vertically and horizontally partitioned data.

Rademacher observations and supervised learning: suppose without loss of generality that we have one sample \mathcal{S} fully labeled. (Linear) Supervised learning proceeds with the minimization of a specific *loss* function, which can be for example the logistic loss (logloss):

$$F_{\log}(\mathcal{S}, \theta) \doteq \frac{1}{m} \sum_i \log(1 + \exp(-y_i \theta^\top \mathbf{x}_i)) . \quad (1)$$

It was recently remarked that this task is equivalent to the minimization of an *equivalent* task over a different kind of input known as *Rademacher observations* (rados): letting $\Sigma_m \doteq \{-1, 1\}^m$ and $\sigma \in \Sigma_m$, the rado π_σ with signature σ is $\pi_\sigma \doteq (1/2) \cdot \sum_i (\sigma_i + y_i) \mathbf{x}_i$ (informally, *one* rado is a sum of the edge vectors of examples over a subset of data). We insist on the fact that this equivalence is tight: $\exists h : \mathbb{R} \rightarrow \mathbb{R}$ strictly increasing and another loss, called the exponential rado loss, $F_{\exp}^r(\mathcal{S}, \theta, \mathcal{U})$, *defined over rados* (and not examples) where $\mathcal{U} \subseteq \Sigma_m$ defines a subset of all 2^m rados, such that

$$F_{\log}(\mathcal{S}, \theta) = h(F_{\exp}^r(\mathcal{S}, \theta, \Sigma_m)) , \quad (2)$$

and this holds *regardless* of classifier θ and training sample \mathcal{S} . Even more, this holds for *other* losses as well (such as the square loss and unhinged loss) and for *regularized* losses as well [Nock, 2016, Patrini et al., 2016b]. It is not our objective to detail this general equivalence, which can be found in the related papers. For the purpose of our setting, one may keep in mind that (i) rados allow to learn in a distributed environment without carrying entity matching [Patrini et al., 2016b], and (ii) rados bring additional privacy guarantees for examples that supervised learning from examples alone may not bring [Nock et al., 2015], depending on the side information available to an attacker. For example, (a) it is NP-hard to answer whether a given set of rados was crafted from a specific subset of examples; (b) It is *geometrically* hard (protection holds regardless of the computational power) to reconstruct examples from a set of rados, even knowing an approximate size of the examples data.

3 Our approach

Even without privacy requirements, the main challenges to our distributed learning setting is that (i) entity matching is not available and (ii) labels are not available everywhere. We have combined the ingredients described in the preceding Section to obtain a system summarized in Figure 1 (left). To be more specific, the system implements the following features:

¹<https://github.com/NICTA/javallier>

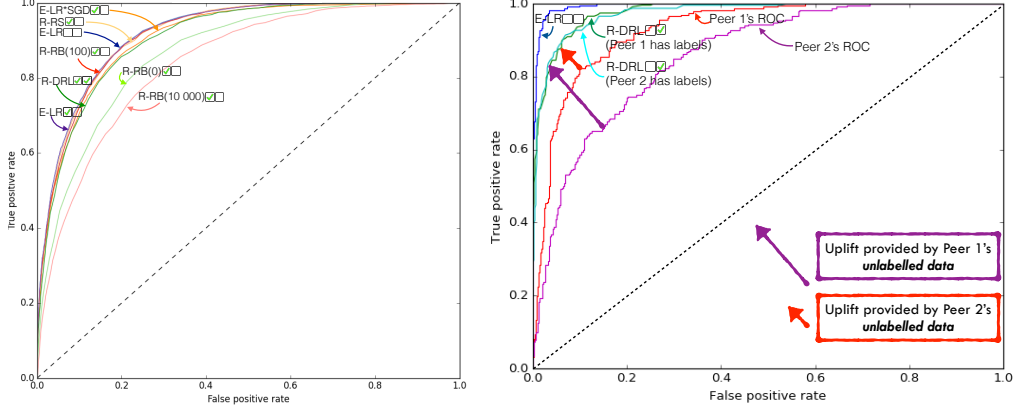


Figure 2: Experiments run on our simulated data (left) and UCI splice (right, see text for details).

- the complete set of operations that go from the weakly labeled distributed datasets to the output of a classifier are *all* performed in the encryption of Paillier cryptosystem;
- the process includes the following steps, in this order: (i) fitting the labels by the unsupervised peers using LLP (Learning from Label Proportions, Patrini et al. [2014]), where the label proportions are encrypted; (ii) crafting of rados by peers, sent to the learner that builds the training rado sample [Patrini et al., 2016b], (iii) learning classifier θ from the rado sample.

Figure 1 (center + right) summarizes the two main steps of supervised learning (computing labels, learning the classifier) in our system, along with the protection guarantees involved.

4 Experiments

We report two simple toy experiments ($p = 2$) with our system. The first is with a simulated dataset ($d = 83$, mixture of Gaussians generated so that Bayes optimal classifier is a linear classifier; the full domain contains 100 000 examples, out of which 25 000 are used for training when full entity matching is available; in this case, the dataset is vertically split between peers). The second is with UCI domain Splice [Bache and Lichman, 2013]. Figure 2 presents our results as ROC curves (the dashed line is random guessing). Each algorithm is indicated as "X-Y" plus two checkboxes. X indicates which kind of data were used, E for examples and R for rados. The two checkboxes indicate, when checked, whether the full process is carried out in the encrypted domain (left) and distributed (right). Y is the type of algorithm used: LR means logistic regression (SGD means stochastic gradient descent, otherwise gradient descent is used); RB is RadoBoost [Nock et al., 2015, Nock, 2016] (the number in parenthesis indicates regularization strength ω , Nock [2016]); DRL is the Distributed Rado Learn algorithm of Patrini et al. [2016b] ($p = 2$ peers). Regularizations used is Ridge. For example, "R-DRL[✓][✓]" indicates DRL, as in Patrini et al. [2016b], but fully run in the Paillier encryption domain. On UCI splice, we focus on DRL and depict performances of each peer to show the leverage of DRL depending on who has labels (LLP follows [Patrini et al., 2014]; first ten features shared, rest split evenly among peers). Globally, results demonstrate the uplift that can be obtained on the performances of one dataset using the contribution of other datasets, even without entity resolution.

5 Conclusion

We have presented a system which is able to learn linear separators from distributed data, with the following key features: from LPP up to all computations take place in Paillier's homomorphic encryption domain, datasets are distributed and they can be weakly labeled. Moreover, the system makes use of the recently introduced concept of Rademacher observations to kill two additional birds in one shot: add further protection layers to encryption, and learn without requiring the parties to perform entity resolution / matching on the distributed data. Because Paillier encryption is additively homomorphic, it makes it particularly easy to craft Rademacher observations in the encrypted domain. Our system illustrates the potential of Rademacher observations for private multiparty learning.

References

- K. Bache and M. Lichman. UCI machine learning repository, 2013.
- R. Bost, R.-A. Popa, S. Tu, and S. Goldwasser. Machine learning classification over encrypted data. Cryptology ePrint Archive, Report 2014/331, 2014.
- R. Canetti. Security and composition of cryptographic protocols: A tutorial. Cryptology ePrint Archive, Report 2006/465, 2006.
- N. Dowlın, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *33rd ICML*, pages 201–210, 2016.
- C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. & Trends in TCS*, 9:211–407, 2014.
- M. Enserink and G. Chin. The end of privacy. *Science*, 347:490–491, 2015.
- O Goldreich. *Foundations of Cryptography*. Cambridge University Press, 1987.
- D.-L. Gorf. Balancing privacy versus accuracy in research protocols. *Science*, 347:479–480, 2015.
- D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *37th ACM SIGMOD*, pages 193–204, 2011.
- D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. on Database Systems*, 39:3, 2014.
- R. Nock. On regularizing rademacher observation losses. In *NIPS*29*, 2016.
- R. Nock, G. Patrini, and A. Friedman. Rademacher observations, private data, and boosting. In *32nd ICML*, pages 948–956, 2015.
- P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.
- G. Patrini, R. Nock, P. Rivera, and T. Caetano. (Almost) no label no cry. In *NIPS*27*, 2014.
- G. Patrini, F. Nielsen, R. Nock, and M. Carioni. Loss factorization, weakly supervised learning and label noise robustness. In *33rd ICML*, pages 708–717, 2016a.
- G. Patrini, R. Nock, S. Hardy, and T. Caetano. Fast learning from distributed datasets without entity matching. In *25th IJCAI*, pages 1909–1917, 2016b.