École Polytechnique

BACHELOR THESIS IN COMPUTER SCIENCE

Privacy-Preserving Mobility Record Linkage Methods for Cross-Silo Vertical Federated Learning

Author:

Cyrus PELLET, École Polytechnique

Advisor:

Guillaume HOLLARD (CREST), Jesse READ (LIX)

Academic year 2022/2023

Abstract

Cross-Silo Federated Learning (CSFL) is an emerging paradigm whereby multiple organizations collectively contribute to the remote training of machine learning models. We are interested in the problem of Record Linkage (RL), concerning how similar entities across distributed datasets can be identified, while adhering to strict privacy standards associated with handling Private Identifiable Information (PII). We focus specifically on the problem of geospatial mobility data linkage; identifying spatially-identified records across distributed datasets. This paper provides a survey of Privacy-Preserving Record Linkage (PPRL) methods, as well as a new protocol adapted for mobility linkage in the context of CSFL. We demonstrate the correctness, privacy and efficiency guarantees that this protocol provides, and demonstrate its practicality by executing it in the context of a full-scale Federated Learning scenario.

Contents

List of Algorithms

1 Introduction

Federated Learning (FL) is a distributed machine learning framework whereby a global model is trained from decentralized datasets. Training takes place in rounds, through which clients train their local models using their private datasets and derive local updates. Such local updates are then aggregated into a new global model by a host party. On the basis of party characteristics and the scale of model training, federated learning approaches can be classified into two types: Cross-Device (CD) FL in which a large number of small distributed entities communicate small amounts of training data, and Cross-Silo (CS) FL, where a small set of organizations or companies participate in the entire training process with very large datasets [18] .

Practical uses of CSFL span multiple domains, including facilitating data sharing within agri-tech supply chains [9] and on-device personalization tuning deployed by large technology companies [35] . The recent focus in CSFL research is motivated by the privacy guarantees its implementations can provide over classical machine learning approaches [17] , though several key challenges and open issues remain to be comprehensively addressed [18] .

A majority of existing research in FL focuses on horizontal data partitions in which parties involved provide records of different entities for the same feature. Nonetheless, there are now abundant real-world cases in which the distribution of data across clients fits the vertical partition (or feature partition) setting, in which parties provide different features for a common set of entities [31] . This class of federated learning settings is dubbed Vertical Federated Learning (VFL).

Several existing studies differentiate two critical stages of the VFL process: Record Linkage (RL) and model training [48] . Record linkage consists in identifying matching pairs of records across feature-partitioned clients. This allows the subsequent training stage to work on distributed (though linked) data records. This is trivially accomplished in the case whereby clients use consistent universally unique identifiers as features – when an entity has the same identifier across all datasets in which it possesses a record – although we shall assume that this is not the case, such as in many real-world scenarios.

In many of the aforementioned sectors of interest, the usage of VFL is greatly motivated by its privacy guarantees, as different organizations want to keep sovereignty of their data throughout the whole training process. In Chapter 5, we provide a detailed scenario in which this is the case. Due to this privacy constraint, we are specifically interested in record linkage protocols which do not leak any information about non-matching records between peers: Privacy-Preserving Record Linkage (PPRL) protocols [14] .

In Chapter 2, we provide definitions for common notions used throughout our analysis, and define a common set of required characteristics, including correctness, privacy, efficiency and utility. In Chapter 3, we provide an overview of existing PPRL methods in the context of Cross-Silo Vertical Federated Learning (CS VFL), evaluating their capabilities against the framework defined previously. We provide novel estimations of the impact of such methods on the resulting accuracy of the federated training process. In Chapter 4 ,we devise an original approach for the specific case of PPRL for mobility data, in the context of a bespoke research project of poverty forecasting in Senegal. We discuss its efficiency improvements over current approaches in the mobility linkage setting, thanks to some experimental results in Chapter 5, and we present some possible refinements. Finally, we give an overview of open issues and future directions of study.

2 Preliminaries

We will begin our study by providing formal definitions for important concepts. We first formally define the VFL context in which our record linkage approaches will evolve, and we then move on to PPRL itself, including formal descriptions of the four desiderata we will be considering throughout our survey of existing methods.

2.1 Vertical Federated Learning

Presume two parties Alice A and Bob B want to collectively train a machine learning model while keeping their respective datasets D_A and D_B private. Denote by $\mathcal{D} := D_A \cup D_B$ their data domain, feature spaces as $\mathcal{X} := X_A \cup X_B$, label spaces as $\mathcal{Y} := Y_A \cup Y_B$ and sample ID spaces as $\mathcal{I} = I_A \cup I_B$. Thus, denote by $D_A := \{X_A, Y_A, I_A\}$ and $D_B := \{X_B, Y_B, I_B\}$ the datasets owned by Alice and Bob respectively. In vertical FL, where we specifically assume that data is partitioned by

features, the following hold specifically [21] :

$$X_A \neq X_B, \quad Y_A \neq Y_B, \quad I_A \cap I_B \neq \emptyset \tag{1}$$

We further assume N common samples $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$ are available to train a joint machine learning model with at least one of the parties holding label information $y_i, \forall i$. Each feature vector $x_i \in \mathbb{R}^{1\times d}$ is distributed amongst the parties A and B, so that $x_{i,A} \in \mathbb{R}^{1\times d_A}$ and $x_{i,B} \in \mathbb{R}^{1\times d_B}$, where d_A and d_B are the feature dimensions of parties A and B respectively. Without loss of generality, assume party A owns the labels as the *host* party, while B shall be the guest party, hence it can be assumed that $Y_B = \emptyset$. The goal of a VFL algorithm is to collectively train a joint model using $\mathcal{D}$ while preserving the privacy of local data and models. The loss of VFL can be formalized as follows [50] :

$$\min_{\theta} \ell(\Theta; \mathcal{D}) := \frac{1}{N} \sum_{i=1}^{N} f(\Theta; x_i, y_i) + \lambda\Omega(\Theta) \tag{2}$$

where Θ denotes the model, f the loss function, and $\lambda\Omega(\Theta)$ the regularizer term. We can decompose Θ into local models Θ_k parameterized by $\theta_k, k \in \{A, B\}$, which operate only on local data, and a global model Θ^G, parameterized by ψ^G, which is only accessible by the host A. We rewrite the loss function as

$$f(\Theta; x_i, y_i) = \mathcal{L}(\Theta^G(\psi^G; \Theta_A(x_{i,A}, \theta_A), \Theta_B(x_{i,B}, \theta_B)), y_{i,A}) \tag{3}$$

where $\mathcal{L}$ denotes the cross entropy loss.

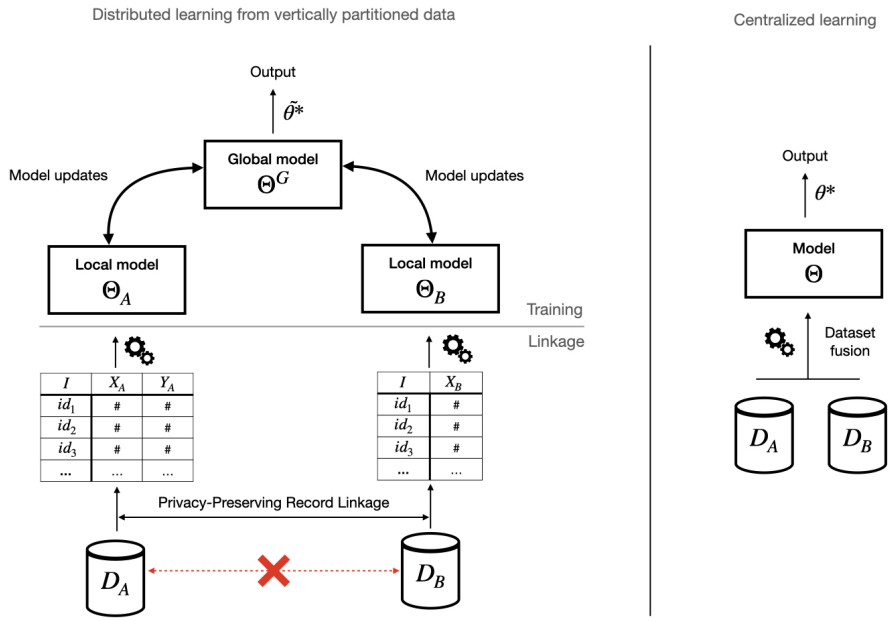


Figure 1: Illustration of a VFL architecture in which local models contribute updates to an isolated global model, compared with a traditional machine learning scenario whereby dataset fusion is performed locally.

2.2 Privacy-Preserving Record Linkage

We start from the preceding scenario, and further consider Alice and Bob to be semi-honest adversaries (def 6). Both parties want to discover all common entities for which they each individually have an associated sample (which we shall refer to as *record* in this context), in order to perform model training. To aid this task, we assume there exists a set of common features $\{x_i^C\}_{i=1}^{k\leq\min(m,n)}$ such that $\{x_i^A\}_{i=1}^{m} = \{x_i^C\}$ We assume feature-induced or record-induced

heterogeneity across D_A and D_B: no two records from either of the two datasets are identical[1]. We let D_A and D_B come from some domain $\mathcal{D}$, with records belonging to some domain $\mathcal{R}$. Moreover, suppose that Alice and Bob have agreed in advance on a collection of common features $X^C := \cap_{k \in A,B} X_k \neq \emptyset$ to help them determine whether records match across their datasets. To designate any two records as "matching" records, the two parties additionally agree on a similarity measure [28] :

Definition 1 (Similarity Measure)**.** A similarity measure $m : \mathcal{R} \times \mathcal{R} \to [0, 1]$ defined on a metric space $\mathcal{M} = (M \subseteq \mathcal{R}, d)$ with a metric $d : M \times M \to \mathbb{R}$ maps two records from a domain $\mathcal{R}$ to a value between 0 and 1. This value is small if records r and s are dissimilar, large if they are close, and 1 if they are identical. We naturally consider that m is commutative, i.e: $m(r, s) = m(s, r), \forall (r, s) \in \mathcal{R}^2$.

Example 2 (Jaccard Similarity Measure)**.** Given two sets P and Q, the Jaccard Similarity is defined as

$$m_j(P, Q) = \frac{|P \cap Q|}{|P \cup Q|} \tag{4}$$

One can trivially verify that m_j satisfies the properties of a similarity measure. We define an associated matching rule I_t^m for the purposes of classifying record pairs as matches or non-matches, outputting 1 for a measure m if its value for any two records r and s exceeds a certain agreed-upon threshold t, and 0 otherwise. Formally, we define the following:

Definition 3 (Record Matching Rule)**.** Denote by $m : \text{Dom}(x_i^C) \times \text{Dom}(x_i^C) \to [0, 1]$ a similarity measure defined over the domain of a common feature x_i^C, and by $t \geq 0$ a matching threshold. The matching rule (or indicator) I is defined as

$$I_t^m(r, s) = \begin{cases} 1 & m(r, s) \geq t \\ 0 & \text{otherwise} \end{cases} \qquad r, s \in \text{Dom}(x_i^C) \tag{5}$$

A record pair $(r \in D_A, s \in D_B)$ is said to be a predicted *match* according to I_t^m if and only if $I_t^m(r, s) = 1$. In complex situations whereby multiple features are taken into account for comparisons, a composite matching rule I_Σ may be defined along with a weight vector to give varying incidence to different compared features across records.

Example 4 (Composite matching Rule)**.** Given two records $r, s \in \mathcal{R}$ with linking attributes $x_1, \ldots, x_n$, we rescale the domain of all similarity measures $m_1, \ldots, m_n$ to $[0, b]$, where $b \geq 0$ and scale attributes according to a weight vector $[w_1, \ldots, w_n]$ agreed upon by Alice and Bob:

$$I_\Sigma = \begin{cases} 1 & \sum_{i=1}^n w_i \cdot \frac{m_i(r.x_i, s.x_i)}{b} \geq t \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

In the context of PPRL, parties Alice and Bob need a protocol $\Pi : \mathcal{D} \times \mathcal{D} \to \mathcal{D}^2$ such that $\Pi(D_A, D_B) := [O_T^A, O_T^B]$, for joint privacy-preserving computation of the linkage between their datasets $D_A \bowtie_{I_t^m} D_B$, where $O_T^A := \{r | r \in D_A \wedge (\exists s \in D_B | I_t^m(r, s) = 1)\}$ and $O_T^B := \{r | r \in D_B \wedge (\exists s \in D_A | I_t^m(r, s) = 1)\}$ for a similarity measure m and threshold t. Besides this output, the only information each party is allowed to infer is the size of the other party's dataset. Any pair of records of D_A and D_B that does not satisfy the matching rule is not to be disclosed to either respective parties.

Much of existing PPRL solutions described in chapter 3 can be commonly decomposed into three main steps [6] :

- Pre-processing and optimizations such as pruning and blocking, which are aimed at reducing the number of comparisons to perform (see 3.1.1).
- Comparisons, aimed at computing a similarity measure $m(r, s)$ for each pair of records $(r, s) \in \mathcal{R}^2$ remaining in the comparison pool.
- Classification, whereby each candidate pair is classified as a predicted "match", or "non-match", which is usually done by computing I_t^m or I_Σ. Supervised machine learning approaches such as decision trees and Support Vector Machines (SVM) have also been used within more advanced classification models [47] .

[1] In situations in which this is not the case, we later describe a secure post-processing step as remedy.

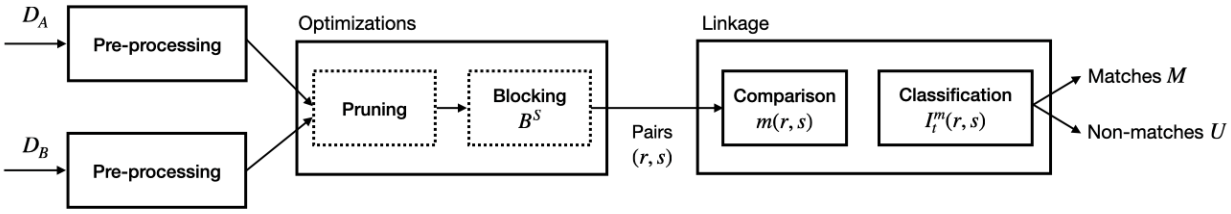


Figure 2: Typical PPRL solution pipeline involving two datasets D_A and D_B, producing sets of record pairs M and U as output.

Although the PPRL problem can be generalized to any number of parties for instance in the interest of cross-device federated learning scenarios, we will focus on the case of two parties, from which most solutions described hereafter can be easily extended to support an arbitrary number of parties. We will however provide computational and communication costs estimates in supporting additional clients for each of the methods described.

We will hence consider that a satisfactory PPRL solution should meet the following criteria:

1. **Correctness**: high precision and recall of matching record pairs [7]
2. **Privacy**: verifiable end-to-end privacy of non-matching records [46]
3. **Efficiency**: computation and communication costs that scale sub-quadratically in the number of input records [1]
4. **Utility**: flexibility in the choice of matching rule (including fuzzy matching and multi-feature weighted matching) and complexity costs incurred in supporting additional clients.

We shall now provide formal definitions of these four desiderata.

2.2.1 Correctness

Let $O_\Pi \subseteq D_A \times D_B$ denote the set of pairs output by the protocol Π as the predicted set of matching records. Π is correct if and only if:

- The protocol returns to both clients A and B the same output O_Π
- $O_\Pi \equiv D_A \bowtie_{I_t^m} D_B$

and incorrect otherwise.

		True linkage	
		Match ($\tilde{M}$)	Non-match ($\tilde{U}$)
Reported linkage	Match (M)	a (true positive)	b (false positive)
	Non-match (U)	c (false negative)	d (true negative)

Table 1: Confusion matrix for the evaluation of record linkage correctness. a, b, c and d represent the number of records in each category and collectively sum up to $|D_A|+|D_B|$.

Consider the linking of D_A and D_B containing $m = |D_A|$ and $n = |D_B|$ records respectively. Subsequently to the execution of Π, each pair of records $(r, s) \in D_A \times D_B$ can correspond to a *match* or a *non-match* (def 3). Thus, barring any optimizations described in section 3.1.1, we can quantify the correctness of each record pair's evaluated matching status as in Table 1, with each of the $m \times n$ compared pairs of records belonging to one of the four cells. The a true positive (correctly linked) record pairs are the pairs belonging to $D_A \bowtie_{I_t^m} D_B$, and should be equal to O_Π in an ideal scenario for Π to be fully correct. Conversely, the d true negative pairs are non-matches that are correctly omitted from from O_Π by Π, and should hence be equal to $(D_A \times D_B) \setminus D_A \bowtie_{I_t^m} D_B$. The two other cells indicate the number of

false matches (b) and false non-matches (c). We will assume that PPRL protocols are designed to produce $O_\Pi = M$ as an output, and we shall explicitly mention specific approaches in which this is not the case.

Existing research has sought to quantify the correctness of a given protocol Π by summarizing a, b, c and d into a single metric, acting as a score on some quality domain, typically $[0, 1]$. Such metrics include:

- *Error rate / Accuracy*: the most widespread accuracy metric in the context of ML, defined as:

$$\varepsilon = \frac{a+d}{a+b+c+d} \tag{7}$$

  Under the assumption that there are no duplicates in D_A and D_B, we have $a \leq \min(m, n)$. As a grows linearly in the total number of records $m + n$, the comparison space $m \times n$ of Π grows quadratically. For this reason, we realize ε is likely to yield senseless results in the context of RL. For instance, we notice that the larger the combined dataset size $m + n$, the lower the error rate for classifying all pairs as non-matches. [PPRL:3]

- *Specificity / Selectivity / True Negative Rate*: used frequently in epidemiological studies, suffers the same problem as error rate due to the fact it does not consider false negatives c, and should not be used alone to quantify RL performance.

$$S = \frac{d}{b+d} \tag{8}$$

- *Precision / Positive Predictive Value*: often seen in RL literature, the fraction of match-classified record pairs that are true positives

$$P = \frac{a}{a+b} = P(\text{true match} \mid \text{predicted match}) \tag{9}$$

- *Recall / Sensitivity / Hit Rate / True Positive Rate*: equally seen in many RL publications, the fraction of true matching record pairs that are classified as matches

$$R = \frac{a}{a+c} = P(\text{predicted match} \mid \text{true match}) \tag{10}$$

  Simply classifying all record pairs as matches yields a perfect recall of $R = 1$, an inconsistency that can be deterred by considering P, which will in this case be low because of b in the denominator.

- *F-measure / F1 Score*: univariate score of the harmonic mean of precision and recall

$$F = 2(P^{-1} + R^{-1})^{-1} = \frac{2a}{2a+b+c} = pR + (1-p)P, \quad p := \frac{a+c}{2a+b+c} \tag{11}$$

  We see that the F-measure is the weighted arithmetic mean with recall scaled to p, and precision scaled to $(1 - p)$. We also notice that the values of p and $(1 - p)$ will vary depending on the linkage method used, since a, b, c and d in Table 1 will vary. Hence, in order to compare linkage methods fairly using F-scores, two separate issues need to be addressed:

  1. Similarity thresholds t for matching rules need to be chosen so that each linkage method in question produces the same number of predicted matches $a + b$.
  2. Similarity thresholds t should additionally be chosen such that the ratio of true matches to predicted matches $\frac{a+c}{a+b}$ reflects the relative importance given to recall and precision.

For the purposes of this study, we shall give equal importance to both misclassifications mentioned above, which entails choosing similarity thresholds t of the different approaches so that

$$\frac{a+c}{2a+b+c} = \frac{a+b}{2a+b+c} \tag{12}$$

This informally corresponds to choosing t so that the number of predicted matches $a+c$ equals the number of true matches $a + b$.

2.2.2 Privacy

We assume that data held by peers is sensitive (possibly involving PII according to definition 17). The execution of PPRL algorithms raises two distinct privacy issues. The first relates to developing mechanisms involved in Π through which the computation of $D_A \bowtie_{I_t^m} D_B$ can be performed without having to partially or fully reveal D_A and D_B. The second issue concerns identifying which computations or sub-procedures lead to information leakages, and what mitigation steps can be undertaken to avoid them.

As part of the protocol Π, client A would like no one else (including B, or any trusted party, such as in the case of many researched approaches [26]) to know whether a specific non-matching record r is present or not in D_A, and analogously for B. This precludes the class of trivial RL solutions where in B sends D_B to A in the clear so that A itself can compute $D_A \bowtie_{I_t^m} D_B$. It equally precludes the large number of proposed solutions in the literature in which a trusted party is involved to process D_A and D_B. In order to evaluate approaches against a common framework, we can differentiate multiple security models to describe privacy guarantees of PPRL methods:

Secure Multiparty Computations (SMC): This second model describes operating environments based on consideration of a real and ideal world scenario. In the ideal world scenario, we assume there exists a fully trustful 3rd party which perfectly computes the given function for our clients and sends back the expected output. In the real world model, such a party does not exist, and clients must compute that same function by themselves by means of message exchanges according to some protocol. A protocol Π is said to be secure if an adversary can learn no more about parties' private inputs in the real world than it could learn in the ideal world, the difference being that no messages are exchanged in the ideal world. Formally, we define the following notion:

Definition 5 (**Indistinguishability of secure two-party computations (IND-S2PC)**)**.** A secure two-party protocol Π that computes some function f is said to be indistinguishable if, for any dataset D_A, and for every pair of datasets D_B and D'_B where $f(D_A, D_B) = f(D_A, D'_B)$, the view of party A during the execution of Π over (D_A, D_B) is computationally indistinguishable from the view over (D_A, D'_B). Formally, for every probabilistic polynomial adversary T,

$$|\Pr\left[T\left(\text{VIEW}_A^\Pi(D_A, D_B)\right) = 1\right] - \Pr\left[T\left(\text{VIEW}_A^\Pi(D_A, D'_B)\right) = 1\right]| < \text{negl}(k)$$

where k is a security parameter which controls the size of the adversary polynomially and the parameterization of the views of the protocol execution, with $\text{negl}(k) = o(k^{-c})$ for all constants c. $\text{VIEW}_A^\Pi(D_A, \cdot)$ denotes the view of party A during the execution of Π.

This definition guarantees that protocols are not allowed to leak any information beyond the size of datasets and the output of f, which, in our case, will often be the output of I_t^m. Furthermore, we consider parties involved to be semi-honest. Some protocols described in literature provide the basis of a solution that is secure against malicious adversaries, which can run any efficient strategy in order to carry out their attack. It is assumed that such adversaries will deviate from the prescribed protocol. We shall not focus on these scenarios, as security against active adversaries typically leads to a reduction in efficiency, introducing the notion of covert security. The task then becomes to ensure that all cheating attempts are consistently detected by peers, which lies outside the scope of our research.

Definition 6 (Semi-honest/honest-but-curious adversary)**.** A party to a joint communication is said to be semi-honest if it does not deviate from the prescribed protocol, but keeps a record of all its intermediate computations.

More formally, we consider a probabilistic polynomial-time functionality (in our case linkage) that we denote $f : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$, where $f = (f_A, f_B)$. A protocol Π exists so that a first party A (with input x) obtains $f_A(x, y)$ and a second party B (with input y) obtains $f_B(x, y)$. Denote as $\text{VIEW}_i^\Pi(x, y, n)$ the *view* of party $i \in \{A, B\}$ during the execution of Π on (x, y) and security parameter n, and equals $(w, r^i; m_1^i, \ldots, m_t^i)$ with $w \in \{x, y\}$ (x if $i = A$ and y if $i = B$), r^i represents the contents of party i's current state (internal tape) and m_j^i represents the jth message it received. We also denote the output of Π's execution by $\text{OUT}^\Pi(x, y, n) = \{\text{OUT}_A^\Pi(x, y, n), \text{OUT}_B^\Pi(x, y, n)\}$.

Now, we say that Π securely computes f in the presence of semi-honest adversaries if there exist probabilistic polynomial-time algorithms S_1 and S_2 such that

$$\{(S_1(1^n, x, f_A(x, y), f(x, y)))\}_{x,y,n} \equiv \{(\text{VIEW}_A^\Pi(x, y, n), \text{OUT}^\Pi(x, y, n))\}_{x,y,n}$$
$$\{(S_2(1^n, y, f_B(x, y), f(x, y)))\}_{x,y,n} \equiv \{(\text{VIEW}_B^\Pi(x, y, n), \text{OUT}^\Pi(x, y, n))\}_{x,y,n}$$

where $x, y \in \{0,1\}^*$ such that $|x| = |y|$ and $n \in \mathbb{N}$.

Differential Privacy (DP): In this privacy model, a given algorithm is considered as private if the distribution of its outputs does not change significantly when one record is changed in the input dataset. We refer to such algorithms for which the total number of records in the input space does not matter as satisfying *unbounded* differential privacy:

Definition 7 (Unbounded (ϵ, δ)-Differential Privacy)**.** We say that a randomized algorithm $M : \mathcal{I} \to \mathcal{O}$ satisfies unbounded (ϵ, δ)-Differential Privacy (DP) if for any output $\omega \in \mathcal{O}$ and any pair of datasets D and D' differing by one record only (i.e: D' can be obtained from D by either adding or removing one record), the following holds:

$$\Pr[M(D) = \omega] \leq e^{\epsilon} \Pr[M(D') = \omega] + \delta \tag{13}$$

DP algorithms are significant in RL as they provide a guarantee that a semi-honest party will be unable to perform attacks in which inducing small changes in their datasets allows them to infer information about their peer's [39].
We observe notably that the IND-S2PC guarantee is equivalent to the Differential Privacy guarantee with $\epsilon = 0$ and $\delta = \text{negl}(k)$, hence all IND-S2PC protocols satisfy unbounded $(0, \text{negl}(k))$-Differential Privacy.

2.2.3 Efficiency

Efficiency concerns in RL stems from the "two Vs" of big data: the challenges of dealing with *Volume* and *Velocity* at scale. The communication and computational costs of the implementation of Π are bounded below by the output size $M = |D_A \bowtie_{I_t^m} D_B|$. We consider linkage scenarios with sub-quadratic output size, and we say that the protocol Π is efficient *volume-wise* if the communication and computational costs are $o(n^2)$. Pruning and blocking techniques described in section 3.1.1 are used to decrease the number of comparisons ahead of linkage, and will be considered independently of the surveyed approaches.

Additionally, considering the *velocity* of data, we recognize applications in which real-time model updates are significant, which literature often qualifies as *Incremental Record Linkage* (IRL) [23] . We aim to significantly improve performance of updates compared to baseline algorithms:

Definition 8 (Velocity performance of incremental linkage algorithms)**.** Consider a PPRL protocol Π acting on datasets D_A and D_B to produce an output O_Π. The computation of Π is said to be efficient *velocity-wise* if the combined runtime over D_A and D_B, followed by Δ_A and Δ_B with $|\Delta_A| << |D_A|$ and $|\Delta_B| << |D_B|$ for some updates Δ_A and Δ_B is faster than the runtime over $D_A + \Delta_A$ and $D_B + \Delta_B$, under the constraint that the F-measure remains equivalent.

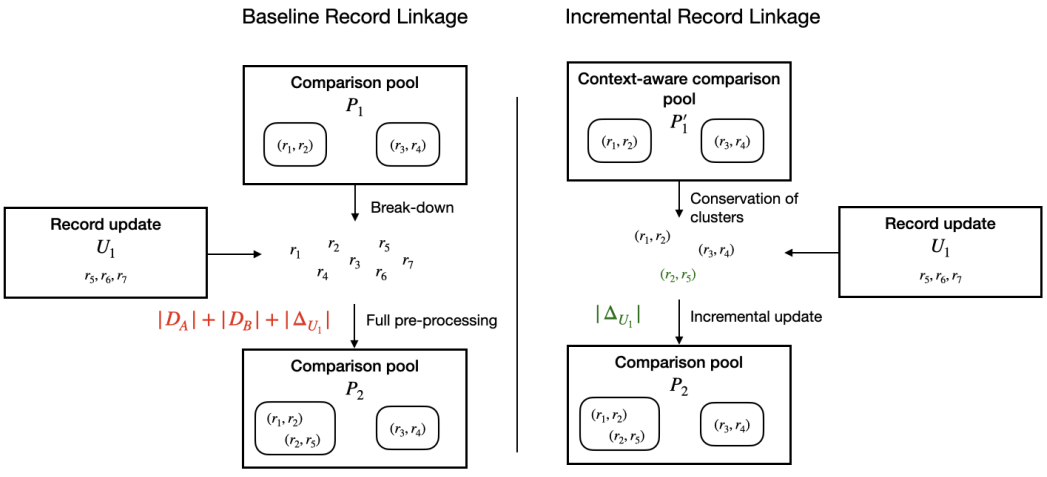


Figure 3: Update process comparison between baseline record linkage methods whereby pre-processing, comparisons and classifications are performed on $D_A \cup D_B \cup \Delta_{U_1}$ and Incremental Record Linkage methods in which incremental computations run with reduced complexity.

2.2.4 Utility

We are specifically interested in using record linkage to train models using FL, hence, in addition to characteristics sufficient to perform linkage between two datasets, we would like PPRL solutions to satisfy the following additional criteria:

1. Flexibility in matching rules: As mentioned previously, we recognize that VFL clients operate on feature-partitioned datasets. Hence, it is seldom possible to perform an exact matching on common identifiers. Depending on whether exact or approximate/fuzzy matching rules are employed in PPRL methods, existing literature differentiates between the following [29] :
   - **Exact PPRL (EPPRL)**: Secure Hash-Encoding comparisons (encoding records using MD5, SHA-1 or other one-way hash encoding functions) and Count-min sketches to identify the frequencies of occurrences.
   - **Approximative PPRL (APPRL)**: Statistical Linkage Keys (SLK) derived from segments of record QIDs, Embedding Spaces to embed QIDs into distance-preserving multi-dimensional metric spaces, Encryption Schemes such as commutative and homomorphic encryption and Bloom Filters.
2. Complexity costs incurred in supporting additional clients: we recognize the need for scalability in the number of parties involved in the PPRL process. For instance, protocols relying on symmetric ciphers suffer from obvious efficiency and privacy shortcomings associated with the distribution of n different keys.

3 PPRL Approaches

Prior to describing our solution for spatially-identified record linkage, we outline existing PPRL approaches and assess them against the four desiderata of the previous section.

Approach	Correctness	Privacy	Efficiency	Utility
Naive linkage	✓	✓	✗	✗
PSI	✗	✓	✓	✗
PSI with expansion	✓	✓	✗	✗
DPRL	✓	✓	✓	✗
Bloom Filters	✗	✗	✓	✓
BTB	✓	✓	✗	✓
ML	$\sim$	$\sim$	✓	✓

Table 2: Overview of the fulfillment of desiderata by surveyed PPRL methods

3.1 Naive linkage

This first trivial class of approaches to PPRL (also referred to as All Pairwise Comparisons (APC)) makes use of a single comparison primitive to exhaustively enumerate and classify record pairs in datasets: every single record in D_A is compared with every single record in D_B. The secure comparison primitive may be implemented using hashes [24] for the case in which only exact matches are considered, garbled circuits [3] or asymmetric cryptography approaches such as homomorphic encryption [34] , depending on the complexity of the matching rule. Many such protocols, devised in the early stages of PPRL research (1990s-2000s), rely on a third party to process comparisons on hashed or encrypted records [33].

Correctness: The output O_Π of naive blocking algorithms in EPPRL scenarios is entirely dependent on the choice of similarity metric and matching rule. For this reason, it is theoretically possible to achieve an optimal recall and precision of 1, since if it is assumed that m and I_t^m are correct, every record pair will be correctly classified. In APPRL scenarios, where outputs are probabilistic, misclassifications are again dependent on the leniency of I_t^m and accuracy of m.

Privacy: While at least one party learns about the other's dataset size, one can trivially verify that naive blocking satisfies IND-S2PC (5); whatever a passive adversary may deduce from the execution of Π could also easily be deduced from O_Π and their own private inputs (which they are allowed to have). Due to the absence of involvement of partitions or additional synoptic information in naive protocols, the notion of Differential Privacy does not apply to our analysis.

Efficiency: Naive blocking involves quadratic computational costs to perform $|D_A| \times |D_B|$ secure pairwise comparisons in total. This is unacceptable at scale, inducing runtimes of well over 40 days for 2 datasets with 6,000 records and 5 attributes each. Moreover, the usage of computationally intensive components such as homomorphic encryption exacerbate the complexity of the protocols, precluding their usage at scale.

Utility: As mentioned previously, naive blocking enables APPR and EPPRL usages alike due to the lack of constraints in matching rules for exhaustive pair comparisons, making them very flexible. We realize however that naive protocols do not scale well in the number of clients, as pairwise comparisons will have to be performed $\frac{n(n-1)}{2}$ times for n clients.

3.1.1 Optimization: Pruning

In scenarios whereby D_A and D_B exceed $5,000$ records, it thus becomes impractical to perform $m \times n$ operations over the entirety of the record space, even using current state-of-the-art processors. A commonly employed trivial pre-processing step to deal with this problem involves removing definite non-matches from the comparison pool ahead of the computation of $m(r, s)$ for all record pairs (r, s). This strategy is referred to as *pruning* or *thinning*, and it is not always relevant, since it rests on the assumption that there exists a subset of features enabling a preliminary matching rule I_{pre} to be applied to individual records in order to eliminate a subset, with significantly lesser computational costs than subsequently calculating I_t^m or complex composite matching rules (example 4). For instance, we can consider the following guiding scenario:

Example 9. In some scenario, a list of locations identified by latitude/longitude pairs are to be classified as matches if they point to the same agglomeration. Linkage without pruning requires $n^2 - n$ comparisons for n records (see table below). With pruning as pre-processing we can eliminate record pairs based on a rough thresholding cutoff of the distance between two points:

(a) Comparison space without pruning: 6 record pairs (in blue) are compared in total

Records	42.5,-70.9	42.8,-70.8	13.6,15.3
4.5,-70.9			
42.8,-70.8			
13.6,15.3			

(b) Comparison space after pruning based on coordinate distance, reducing the number of comparisons to two (in blue)

Records	42.5,-70.9	42.8,-70.8	13.6,15.3
4.5,-70.9			
42.8,-70.8			
13.6,15.3			

Table 3: Geographical linkage scenario in which a trivial pruning strategy based on pre-comparison of the distance between lat/lng coordinates can dramatically reduce the number of comparisons during subsequent linkage. In this case scenario, we set $I^t_{\text{pre}}(r, s) = 1$ if $|\text{lat}_1 - \text{lat}_2| < t \cap |\text{lng}_1 - \text{lng}_2| < t$, and 0 otherwise, for some threshold $t > 0$.

With no incurred privacy or utility impact, pruning is an effective pre-processing step for many PPRL approaches to boost performance by reducing the number of comparisons, thought at the expense of correctness if not performed properly.

3.1.2 Optimization: Blocking

More generally, pruning can be extended or complemented with a partitioning procedure to reduce the number of comparisons even further for more complex datasets. We refer to this partition step as blocking. Blocking hence involves pre-classifying likely matches into "blocks" to speed up computations. This again rests on the assumption that a set of "reliable" features (such as country or gender) are available for pre-removal of such pairs.

Definition 10 (Blocking strategy.)**.** Given k bins $\{B_0, \ldots, B_{k-1}\}$, records in D_A and D_B are hashed by a blocking function $\mathcal{B}$ to a subset of the k bins. The set of records in D_A (resp. D_B) falling into the ith bin are denoted by $\mathcal{B}_i(D_A)$ (resp. $\mathcal{B}_i(D_B)$). A blocking strategy $\mathcal{B}^S \subseteq [0, k) \times [0, k)$ specifies pairs of bins of D_A and D_B that are compared during the linkage process.

The computational complexity of blocking approaches vary depending on the complexity of the blocking strategy. For instance, mapping a single numerical value into bins requires $o(n)$ computations, whereas complex matching rules run in

$o(n^2)$ operations. [44] provides a survey of the correctness and efficiency impact of various blocking strategies. Notably, a blocking strategy is said to be sub-quadratic if the number of candidates matches for $n = |D_A| + |D_B|$:

$$\text{cost}_{\mathcal{B}^S}(D_A, D_B) = \sum_{(i,j)\in\mathcal{B}^S} |\mathcal{B}_i(D_A)||\mathcal{B}_i(D_B)| = o(n^2) \tag{14}$$

The aim of blocking strategies is to remove as many record pairs as possible from the set of non-matches U, without removing any pairs from M, as efficiently as possible. We can introduce two complexity metrics to qualify the efficiency and quality of such blocking methods:

1. *Reduction ratio*: measures the relative reductions of the compared record space, without taking into account the quality of the reduction (how many record pairs from M and U are removed in the process)

$$R_r = 1 - \frac{O_{\mathcal{B}}}{m \times n} \tag{15}$$

   where $O_{\mathcal{B}} \leq m \times n$ is the number of record pairs remaining after execution of the blocking strategy $\mathcal{B}$. The smaller the number of pairs $O_{\mathcal{B}}$, the larger the ratio R_r, and the greater the complexity savings in the subsequent linkage process.

2. *Pair completeness*: analogous to recall (2.2.1): describes how many true positives were generated by blocking as a ratio of all true positives.

$$C_{\Pi} = 1 - \frac{a^{\Pi}}{a} \tag{16}$$

   with a^{Π} being the number of true positives yielded by some RL protocol Π after blocking by strategy $\mathcal{B}$, and a being the number of true positives yielded by Π without blocking.

Many existing protocols leveraging blocking, however, do not provide valid end-to-end privacy guarantees [20] [19] [25]. Indeed, so-called *frequency attacks* allow an adversary to match the frequency distributions of blocks in an encoded dataset with the frequency distribution of records from another known dataset [49]. Methods proposed to remediate this issue include perturbing blocks in such a way that they become differentially private. Chapter 3.3 outlines such a method.

Lastly, blocking algorithms often involve similarity measure computations prior to linkage [27] [25] , and lack operational capabilities for efficient matching of multiple large datasets [15] [22]. Recent developments such as bit tree blocking [37] enable multi dataset linkage using Bloom Filter-encoded records inserted in bit trees such that similar records are grouped in common leaves, or Frequent Pairs Schemes (FPS), in which only record pairs achieving high LSH collision rates are compared, though on a process highly dependent on parameter tuning.

3.2 Private Set Intersection (PSI)

This second category of approaches was originally devised to solve a specific case of PPRL: finding the intersection of two sets without revealing any information about elements that are not in the intersection. PSI algorithms have been used in scenarios whereby two parties want to perform "join" operations over database tables they must keep private, in functionalities such as genome alignment or relationship path discovery [13]. For this reason, although these algorithms have sub-quadratic complexities, they only function with equality predicates, rendering them intolerant to errors and inflexible in the choice of matching rule [11] .

The most basic form of PSI protocols works as follows: Alice and Bob agree on a collision-resistant hash function H. Alice then sends the set $D_A^* = \{H(x)|x \in D_A\}$ to Bob. Bob computes the output as $D_A \cap D_B := \{y \in D_B|H(y) \in D_A^*\}$. Since $H(\cdot)$ is collision-resistant, the entropy of D_A^* will be the same as for D_A. This type of approaches can only be qualified as secure with large domains with high entropies, without which an adversary can easily perform brute-force attacks whereby the hash function can be applied to all items that are likely to be in the input set. A variant of this basic approach makes use of Horner's rule and cryptographic polynomial to evaluate roots corresponding to intersecting elements [11] .

Secure variants such as Diffie-Hellman PSI (DHPSI) can be implemented using elliptic curves or finite field exponentiation, providing additional security guarantees:

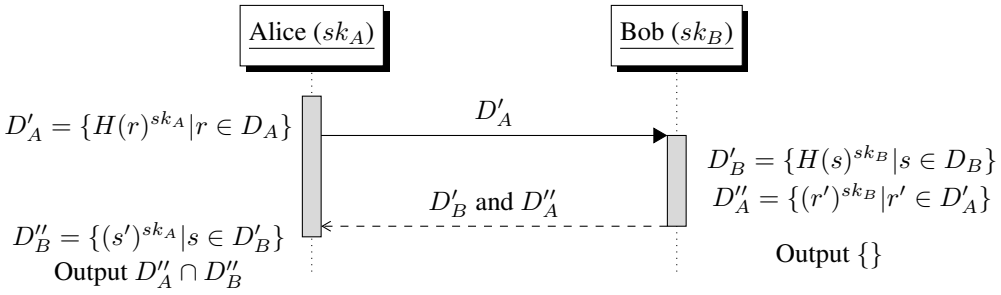


Figure 4: Sequence diagram of one-sided DH-PSI, whereby Alice and Bob encrypt their records with their respective private keys sk_A and sk_B.

Correctness: PSI methods have been shown to achieve poor recall for general matching functions, surveyed extensions solely allow composite exact matches on compared features. Conjunctions and disjunctions on collections of attributes cannot be performed.

Privacy: Considering semi-honest adversaries, the security of such a protocol is based on the irreversibility of the hash function and the hardness of the decisional and one-more Diffie-Hellman assumption (18). Much like naive methods, PSI methods ensure IND-S2PC; the views of Alice and Bob in PSI protocols can be simulated given only their input and output to the protocol [51]. According to our definition of correctness in 2.2.1, we would like Bob to know about O_Π. The mutual DH-PSI variant includes an additional round in which Alice sends D''_B to Bob so that it can also compute the intersection, however, an eavesdropper in this case learns both D''_A and D''_B, hence it can compute $|D''_A \cap D''_B|$. Overall, DH-PSI is very susceptible to man-in-the-middle attacks, though recent research proposed updated protocols that achieve stronger privacy guarantees [40].

Efficiency: With such approaches, we obtain a computational cost of $O(|D_B| \cdot \ln\ln|D_A|)$, which is sub-quadratic in $n = \max(|D_A|, |D_B|)$. Recent contributions using Oblivious Transfer (OT) focus on reducing this computational cost further [36] , and circuit-based methods reduce communication overheads, which are now linear [51].

Utility: PSI methods are limited to EPPRL scenarios, making them inflexible in the choice of matching rule. However, for small datasets, the technique of *expansion* can be utilized:

3.2.1 Extension: Fuzzy matching with expansion

In order to remediate the lack of flexibility in matching rule choices for PSI, techniques of expansion has been described in literature. Using such a setup allows PSI protocols to achieve high recall for general matching rules, enabling fuzzy matching applications. Expansion is implemented by implementing so-called expansion strategies:

Definition 11 (Expansion strategy.)**.** Given datasets D_A and D_B with a common domain $\mathcal{D}$ and a matching rule I_t^m, an expansion strategy $\mathcal{E}_I$ adds all records $r' \in \mathcal{D}$ to D_A for all records $r \in D_A$, such that $m(r, r') \geq t$ to obtain an expanded database D_A^x.

Once D_A^x has been obtained, PSI methods can be applied on D_A^x and D_B to yield the desired output $D_A \bowtie_{I_t^m} D_B$, while still satisfying the IND-S2PC privacy guarantee. However, depending on the leniency of the matching rule I_t^m and the number of rules to expand, $|D_A^x|$ can be orders of magnitude larger than $|D_A|$, making expansion critically inefficient in the size of D_A. An additional challenge relating to the use of complex matching rules is to generate all matching pairs for a record. For instance, if the matching function m can encode boolean 3-CNF formulas, finding all values of r' such that $m(r, r')$ for some record r could be impossible [16] . Enumerating a superset of matches further increases the computational cost of such an approach.

3.3 Differential Privacy Record Linkage (DPRL)

Differential Privacy has often been described as one of the state-of-the-art provable privacy definitions for sharing sensitive data. Having been successfully adopted by companies and government entities alike, it enables the computation

of aggregate statistics from a dataset without revealing the presence or absence of a specific record in the dataset. An algorithm is said to satisfy differential privacy if adding or removing a record in its input set does not significantly affect its output. In Definition 7, we characterized the notion of differential privacy, though many protocols in literature lack formal end-to-end privacy guarantees [20] [19]. To remediate this issue, recent advances introduce the notion of Output-Constrained DP (OCDP) [16] [30]. This notion addresses the need to reveal records belonging to $D_A \bowtie_{I_t^m} D_B$ for the successful execution of a protocol Π, the ability to reveal statistics about records outside O_Π and the preclusion of any leak about the presence or absence of individual non-matching records in D_A or D_B:

Definition 12 (Output-Constrained Differential Privacy (OCDP))**.** A PPRL protocol Π for computing a function $f : \mathcal{R} \times \mathcal{R} \to \mathcal{O}$ is said to satisfy $(\epsilon_A, \epsilon_B, \delta_A, \delta_B, f)$-OCDP if for any pair of databases (D_B, D'_B) such that

1. $f(D_A, D_B) = f(D_A, D'_B)$
2. $D_B \setminus D'_B \cup D'_B \setminus D_B \neq \emptyset$ and
3. $\nexists D_C \in \mathcal{R}$ with $f(D_A, D_B) = f(D_A, D_C)$ such that $(D_B \setminus D_C) \cup (D_C \setminus D_B) \subset (D_B \setminus D'_B) \cup (D'_B \setminus D_B)$

the views of Alice during the execution of Π to any probabilistic polynomial-time adversary T satisfies

$$\Pr[T(\text{VIEW}_A^\Pi(D_A, D_B)) = 1] \leq e^{\epsilon_B} \Pr[T(\text{VIEW}_A^\Pi(D_A, D'_B)) = 1] + \delta_B \tag{17}$$

which also holds for the views of Bob with ϵ_A and δ_A.

3.3.1 The Laplace Protocol (LP)

The first DPRL algorithm we consider, dubbed the Laplace Protocol (LP), consists in inserting a specific number of fabricated records in each of the k bins of a blocking strategy $\mathcal{B}^S$ so that the sizes of each bin become differentially private. These fabricated records have no incidence on our output $D_A \bowtie_{I_t^m} D_B$ since they are specifically crafted to match no other record (for instance by taking on out-of-bounds or incompatible values which still appear nominal to the other party after hashing or encryption). Its steps can be summarized as follows

1. Alice and Bob hash their records into sets of blocking bins $\mathcal{B}(D_A)$ and $\mathcal{B}(D_B)$ according to an agreed-upon blocking protocol $\mathcal{B}$.
2. The number of records in the bins are then increased according to a truncated Laplace distribution, such that they satisfy (ϵ, δ)-DP. Concretely, this means that η dummy records are added to each bin, where η is sampled from a random variable following a Lap$(\epsilon, \delta, \Delta\mathcal{B})$ distribution. Added records lie in an extended domain in such a way that they will not match with any other records in the true domain.
3. Alice and Bob perform comparison and classification steps to compare records in $\tilde{\mathcal{B}}_i(D_A) \times \tilde{\mathcal{B}}_j(D_B)$ for $(i, j)\mathcal{B}^S$. Much like in naive linkage, secure matching can be implemented using garbled circuits or (partially) homomorphic encryption.

Correctness: Compared with naive linkage methods, no records are pruned, and dummy records lie in a match-prohibitive extended domain. Hence, the output of the LP protocol will be identical to the naive linkage output.

Privacy: LP satisfies $(\epsilon_A, \epsilon_B, \delta_A, \delta_B, f)$-OCDP [16].

Efficiency: We have that the asymptotic complexity of LP is sub-quadratic. *Sort & Prune* and *Greedy Match & Clean* optimizations further decrease the number of comparisons performed by Alice and Bob [16], up to 50% in some cases.

Utility: LP offers the same flexibility in matching rules as naive linkage, enabling APPRL and EPPRL applications. However, in the case of flexible matching rules, dummy records must be constructed in such a way to avoid accidental linkages in the real domain. Nonetheless, DPRL protocols suffer from critically large comparison spaces with increasing number of datasets, even when blocking and pruning methods are utilized. This exponential growth in computation and communication costs make DPRL protocols unsuitable for applications involving more than two clients.

3.4 Probabilistic methods

Probabilistic Data Structures make use of space-efficient representations of data in order to (approximately) respond to queries about such data [8]. Amongst those, Bloom filters and Count-Min sketches have notably been used in the context of PPRL protocols to indicate whether an entity is possibly present in a given set (in our case the intersection of datasets D_A and D_B.

The first formulation of an efficient multi-party PPRL approach using Bloom filters was introduced in [42] and only supported exact matching. An updated method by [41] built upon this work to include fuzzy matching:

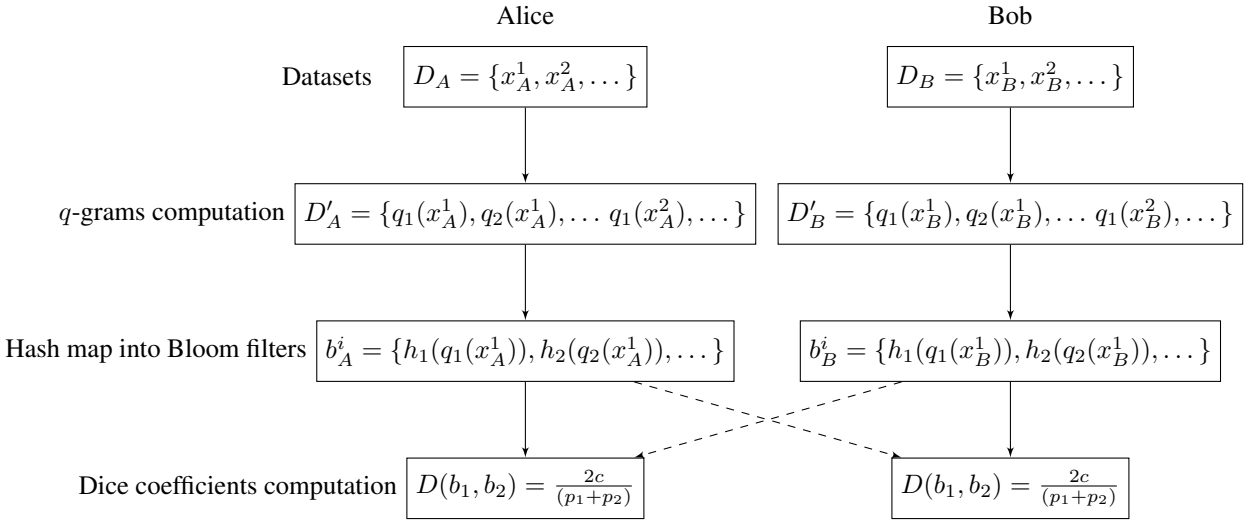


Figure 5: Bloom Filter-based approximate matching PPRL approach. $q_i(r)$ denote the ith q-gram of a given record r. $h_1, \dots, h_k$ are k independent hash functions, c denotes the number of bit positions set to 1 in the logical AND operation between Bloom filters b_1 and b_2, and p_i denote the number of bit positions set to 1 in b_i, $i \in \{1, 2\}$.

Although Bloom filters provide good performance results on large datasets, they engender for a (tunable) false positive rate, inducing false positives in the linkage output. Moreover, such structures have been shown to be vulnerable to a range of *cryptanalysis attacks*, allowing an adversary to recover the set of possible records for a given Bloom filter. For Bloom filter-based blocking protocols, small bins enable frequency-based attacks whereby frequently occurring filters can be linked back to commonly occurring values in the dataset [**?**].

A recent area of interest surrounding Bloom filters has been their use within efficient blocking approaches. On particular method, dubbed *Bit Tree Blocking (BTB)*, makes use of a single-bit tree into which bloom-encoded records are placed, before being split recursively based on selected bit positions. Recent publications detail the use of multi-bit trees instead to improve accuracy [38] :

3.4.1 Bit Tree Blocking (BTB)

Surveyed protocols until this point suffer from exponential complexity in terms of the number of parties, precluding them from satisfying sufficient utility criteria when increasing the number of participating parties. Advanced blocking and indexing techniques such as Sorted Neighborhood Indexing, Q-gram indexing and Canopy Clustering [5] aim to solve this issue by treating blocking as a multi-party protocol and using more efficient clustering methods. Multi-bit indexing can be summarized as follows:

1. Bloom filters are split into mini-blocks
   (a) Bloom filters are generated for records in the datasets and added to a queue $\mathcal{Q}$ as a single block.
   (b) Parties compute their own *ratio vector*, containing the ratio between the number of 0s and 1s for each bit position in their Bloom filters.

(c) An extended secure summation protocol computes common bit positions suitable for block splitting. Bit positions with a sum smaller than a certain threshold are selected into a set of splitting bit positions. For all possible bit combinations, a common best bit combination is determined, and used to split the current block into mini-blocks.

(d) The process is performed again iteratively until all mini-blocks contain a number of records within a pre-determined range.

2. Mini-blocks are merged using clustering methods

   (a) The Bloom filter with the highest similarity to all other Bloom filters in each mini-block is selected as the block's *centroid*, using a Hamming distance-based similarity computation.

   (b) Blocks are merged based on centroid distance until the size of resulting clusters reaches a pre-defined threshold.

3. Each resulting cluster is used as a block within the comparison step of the PPRL pipeline.

Correctness: BTB methods provide improved recall over more basic blocking techniques, though performance is dependent on the choice of value for ms_{max}, the target number of records per merged blocks. Dataset size and the number of parties must be taken into account to maximize recall and efficiency while guaranteeing sufficient privacy.

Privacy: The added component of a secure summation protocol for exchanging Bloom filter ratio values between parties satisfies IND-S2PC, moreover, neither parties are capable of deducing anything about each other's input [38] .

Efficiency: According to the authors of BTB, the full protocol has a communication cost of $O(P \cdot \log_2(n))$ for P parties, which is significantly lower than previously surveyed methods. Computational complexities for the different stages of BTB can be summarized as follows: $O(k \cdot n \cdot N)$ for Bloom filter generation (k hash functions, N records and n q-grams per record), $O(N \cdot \log_2(N/s_{max})/d_{max})$ for splitting into mini-blocks and $O(C^2)$ for the merging of mini-blocks into C clusters.

Utility: Thanks to low communication costs and good performance on large datasets, BTB is suitable for multi-party scenarios. Subsequent linkage steps can be chosen so that APPRL and EPPRL operations may take place, hence, BTB can be considered as flexible in the choice of matching rule, provided blocking criteria are set accordingly.

3.5 ML linkage

Machine Learning (ML) can be implemented to aid various stages of the linkage process, in particular for scenarios in which matching rules cannot be easily defined. These types of algorithms, however, introduce a new class of vulnerabilities related to *membership inference attacks*, which provide information about a target record in the training dataset by querying the model [43] .

Supervised Meta-Blocking: a post-processing step to blocking whereby a block collection is fed into a supervised binary classifier that labels pairs in the comparison pool as highly likely to match or not. Pruning can then be performed on the resulting labeled blocks [12]. Support Vector Machines (SVM) and decision trees have been often cited as fitting supervised learning techniques employed for record linkage. Unsupervised techniques such as K-means clustering and Expectation Conditional Maximization (ECM) have also been cited as relevant in the context of RL .

Bloom filter encodings: Recent research [47] identified Bloom filter-encoded data as being insusceptible to membership enference attacks. Hence, since similarity measures are conserved in Bloom filter encoding spaces, machine learning models for classification, clustering or regression can be fed Bloom-encoded data as inputs. In PPRL, this is useful in scenarios where supervised models are trained as drop-in replacements for matching rules I_t^m.

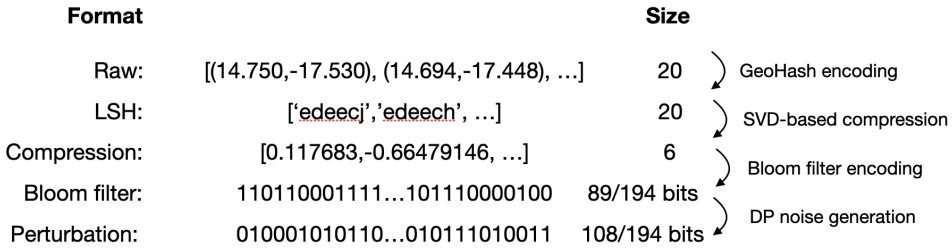


Figure 6: Bloom encoding for ML tasks: geo coordinates are hashed using a LSH function (in this case GeoHash), before being compressed by a dimension-reducing method such as PCA, t-SNE or SVD. The resulting list is then encoded into a Bloom filter space, which is rendered private by performing a XOR operation on differentially private noise.

Correctness, Efficiency: Correctness and efficiency analysis of Machine Learning approaches deserve a full study of their own, nonetheless, we consider a few related pitfalls and limitations in the context of record linkage:

- Training data is not always available in record linkage scenarios, even more so in privacy-preserving settings (5.1).
- In scenarios whereby model updates are performed (for instance in VFL settings), precautions must be taken to minimize the impact of real and concept drift on the linkage process. This can involve retraining the model with new data or adjusting hyper-parameters to maintain performance over time.
- With any ML model, good feature selection is essential to achieve sufficient accuracy and efficiency. Including irrelevant or redundant features can slow down the algorithm and reduce accuracy, while not including enough relevant features can result in missed matches.

In the context of Bloom filter encodings, [47] uncover a number of valuable observations: lower Bloom embedding dimensions lead to higher classification accuracies, so do larger Bloom filters. Unsurprisingly, increasing the noise value used for perturbations decreases accuracy, and sub-optimal values for k, the number of hash functions, lead to reduced accuracy due to increases in false positives.

Concretely, methods such as Randomized Aggregatable Privacy-Preserving Ordinal Response (RAPPOR) [10] are used to determined which bits are to be perturbed from a privacy parameter f depending on a privacy budget $\epsilon = \frac{2n\ln(2)(1-f/2)}{f/2}$, where n is the number of records hash-mapped into bloom filters. The perturbed bit value b'_i of a bit b_i in a Bloom filter b is given by:

$$b'_i = \begin{cases} 1 & \text{with probability } f/2 \\ 0 & \text{with probability } f/2 \\ b_i & \text{with probability } 1-f \end{cases} \tag{18}$$

Privacy: Encodings such as the Bloom filter approach described above provide additional privacy guarantees when working with sensitive data. We can nonetheless consider a several attacks aimed at inferring private information in ML PPRL scenarios:

1. Membership inference attacks: an adversary is able to construct so-called *shadow models* that imitate the behavior of the target model to determine whether a given record was part of its training dataset or not, without any knowledge of the model's structure and parameters [43]. The certainty of such inferences is mitigated by the addition of noise perturbation in Bloom filters, decreasing their certainty.
2. Differential attacks: in scenarios whereby adversaries possess an auxiliary dataset containing duplicates of records held in D_A and D_B, analysis of model output under small input differences can reveal relationships between those datasets. Specific records can hence be identified in private datasets using such methods. Nonetheless, differentially private blocking methods and Bloom filter divisions can mitigate such attacks, even with differences of one record (according to definition 7).

Utility: ML linkage methods enable more complex matching scenarios, such as completing records with missing or incomplete data and incorporating various features and parameters that would be difficult to capture using deterministic approaches. Such methods are particularly efficient for fuzzy matching, as models can be trained to recognize synonyms (e.g: linking "123 Main St." with "123 Main Street"). The linkage process itself can be integrated within a VFL framework to efficiently distribute the task across an arbitrary number of clients.

4 Mobility linkage

Geolocation data has recently become an essential component of many machine learning applications. By analyzing patterns in human mobility, machine learning models can make accurate predictions about future events, such as traffic congestion, social mobilizations, or consumer behavior. Geolocation data can also be used to identify anomalies or outliers, such as fraudulent transactions. With the proliferation of geolocation-based sensors and devices, such as smartphones and IoT devices, geolocation data is becoming increasingly available, making it an important area of research and development for machine learning.

Nonetheless, working with geographical data presents unique challenges in privacy protection. Firstly, location data has been shown to be extremely identifiable, even in aggregated settings. Research shows that 4 spatio-temporal Call Detail Record (CDR) traces only are needed to uniquely identify 95% of individuals in a pool of 1.5 million people over a period of fifteen months. Secondly, due to the predictable nature of human mobility, the future whereabouts of an individual can be inferred with a high certainty, despite heterogeneous mobility patterns across a whole population [32]. Hence, working with geographical data calls for the implementation of several measures to ensure privacy constraints are satisfied:

1. **Pseudonymization** pre-processing steps ensure that individual records do not contain PPI data that would allow an adversary to re-trace individuals from known attributes. This includes scrubbing / hashing names, locations and phone numbers into unique identifiers that carry no additional purpose than to uniquely differentiate entities. Nonetheless, pseudonymized data stills offers opportunities for re-identification, in the case where an adversary is able to exploit the correlation between multiple records.

2. **Anonymization** procedures ensure that the connection between records and individuals are fully severed. Such techniques my involve modifying data by salting or skewing values randomly in such a way that cross-record tracing becomes difficult. For instance, while pseudonymized mobility records could still be identified if the visited locations of an individual are known by an adversary, anonymized records of a single individual may be identified by random identifiers, in which case the linking cannot be performed trivially anymore.

We are hence interested in PPRL methods that specifically address the specificities of geographical re-identification concerns, with minimal impact on linkage performance and accuracy. We recognize the following problem definition for mobility linkage:

Problem 13 (Mobility Linkage)**.** We consider a specific case of PPRL scenario whereby D_A consists of **entities** consisting of tuples (i, l), with i a unique identifier and l location coordinates and D_B consists of **records** consisting of tuples (i, l, t), where t represents a timestamp, and the entity identifier i lies in a distinct domain from the ones used for entities in D_A. We assume that $|D_B| >> |D_A|$, and that hence multiple records are to be linked to a single entity.

4.1 Pre-processing

In order to associate a set of records to a given entity, we establish an inference step as pre-processing for D_B in order to identify candidate locations to be matched with locations of D_A. We hence establish a set of (rule-based or supervised ML) algorithms $\{A_1, \cdots, A_N\}$, which, given a set of records $(i_i, l_i, t_i)_{i=1}^{|D_B|}$, produce a set of candidate locations $L_1, \ldots,$ ordered by certainty. Using a collection of such algorithms, we obtain a set $[\{L_1^1, \ldots\}, \cdots, \{L_1^N, \ldots\}]$ of predicted locations to match with entities in D_A.

We then evaluate those predictions using Simple Matching Coefficients (SMC), for algorithms A_A and A_B:

$$\text{SMC}(A_A, A_B) = \frac{\sum_{i=1}^{N} \delta(L_{A_i}, L_{B_i})}{N} \tag{19}$$

where N is the number of individuals, and $\delta(L_{A_i}, L_{B_i})$ equals 1 if the identified locations for user i are identical between algorithms A_A and A_B, and 0 otherwise. The resulting SMC value can be interpreted as the fraction of surveyed users for which both algorithms agree on a result. We can quantify the certainty of predictions by computing a spatial uncertainty measure [45]:

$$\mathrm{SU}_{L_i} = \sum_{j,k,\dots} \frac{p_j \cdot d(L_i, L_j)}{2p_n} \tag{20}$$

where SU_L, given in meters, denotes the spatial uncertainty for detecting a location of interest, L_i, $\{j, k, \dots\}$ denote the considered observations by various algorithms, $d(L_i, L_j)$ denotes the distance between locations L_i and L_j (in m) and p_j and p_n are the number of considered observations of locations L_j and L_n from the different algorithms.

For later stages of the linkage process, we wish to choose a similarity measure such that when two records point to nearby locations, their pair score is similar to one another. Literature presents Locality-Preserving Hash Functions (LPH) as fitting for this use case.

Definition 14 (Locality-Preserving Hash Function (LPH))**.** A LPH (or similarity-preserving hash) function H_ℓ is a hash function that satisfies the following:

$$m(q, r) < m(r, s) \implies m(H_\ell(q), H_\ell(r)) < m(H_\ell(r), H_\ell(s)) \tag{21}$$

for some records $q, r, s \in \mathcal{R}$ and some similarity measure $m : \mathcal{R} \times \mathcal{R} \to [0, 1]$.

For our particular case, we will make use of a coding system dubbed *GeoHash*, which provide a convenient way to represent coordinates as strings with variable degrees of precision depending on a length parameter k (see Table 9). For instance, coordinates $(14.74712421, -17.52570173)$ is encoded as `edeechwh` for $k = 8$.

4.2 Key and candidate cells

Record pruning: We take advantage of the precision levels of GeoHash to perform a preliminary pruning of pairs. We first hash locations with a precision level k_c, and assign to entity in D_B a *primary hash*, the GeoHash cell at level k_c which contains the most events from that particular entity. Entities pairs that do not have their primary hash in common are discarded (since we assume they cannot be the same entity).

Key cells computation: We then compute hashes with a precision level $k_c + \Delta_K$, to then compute a set of N *key cells* S_K for each entity, where most of its activity was recorded. The value of Δ_K can be determined iteratively. Small values will cause false negatives due to the exceeding granularity of the resulting blocking, and big values will limit the performance gains of the blocking phase. A technicality to consider occurs when a surveyed event occurs close to a border (1/10th of the cell's edge width) between two GeoHash cells. In this case, we include both cells in the set of key cells.

Candidate cells computation: While key cells give an indication of frequent activity around a particular location, they do not carry useful meaning for linkage with specific entities in D_A. Hence, we feed S_K into the task-specific algorithms $\{A_1, \cdots, A_N\}$ described previously to produce a reduced set of *candidate cells* S_C. These candidate cells are the ones used in subsequent linkage steps.

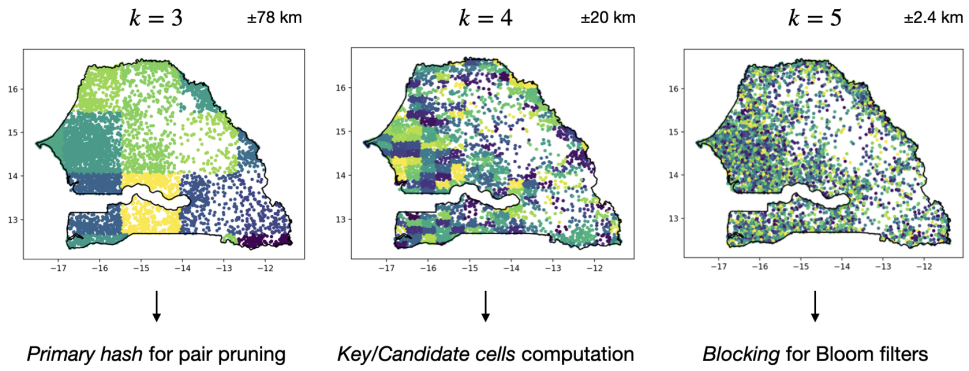


Figure 7: Incremental GeoHash precision levels for geographical cells

Algorithm 1 Candidate cells computation for D_B

Require: $D_B = \{x_i = (i_i, l_i, t_i)\}_{i=1}^{N}$, k_c, Δ_K,S_{max},c_t,$\{A_1, \cdots, A_N\}$
  $\{\mathcal{H}(l_i)\}_{i=1}^{N} \leftarrow$ computeGeoHashes$(D, k_c + \Delta_K)$
  $T \leftarrow \emptyset$
  for $\mathcal{H}(l_i)$ **do**
    $T[i_i] = [\mathcal{H}(l_i)]$
    $c \leftarrow$ computeCentroid$(\mathcal{H}(l_i))$
    if $d(c, l_i) \geq c_t$ **then**
      $c_n \leftarrow$ computeNeighbor$(\mathcal{H}(l_i))$
      $T[i_i]$.append(c_n)
  $S_K, S_C \leftarrow \emptyset$
  for $i \in \{1, \ldots, N\}$ **do**
    $S_K \leftarrow$ k-means$(T[i_i])$
  for $S_K^i \in S_K$ **do**
    for $A_i \in \{A_1, \cdots, A_N\}$ **do**
      $\{L_1^i, \cdots, L_n^i\} = A_i(S_K^i)$
      if SMC$(S_C[i_i], \{L_1^i, \cdots, L_n^i\}) \geq S_{max}$ **then**
        $S_C[i_i]$.append$(\{L_1^i, \cdots, L_n^i\})$
  return S_C

Here, SMC is the Simple Matching Coefficients function described in Section 4.1.

4.3 Bloom Filter Exchange

Once candidate sets S_C have been constructed for every record in D_B, GeoHashes are re-computed to an accuracy level suitable for matching with records in D_A, before being inserted in bins of regional granularity. The number of records in each bin is rendered differentially private thanks to the technique described in section 3.3.1 (Laplace Protocol). Then, Bloom filters are constructed for each bin.Wherever Bloom filters are used, given a number of records n and a target probability of false positives p, parameters are set as follows to minimize the occurrence of false positives:

- The number of bits in the filter: $m = \lceil \frac{n \log(p)}{\log(1/2^{\log(2)})} \rceil$
- The number of hash functions $k = \lfloor \frac{m}{n} \log(2) \rfloor$

For instance, inserting 80,000 records with a desired false positive probability of 0.00000001 (1 in 99857606 records), we set $m = 3067219$ (374.42KiB) and $k = 27$.

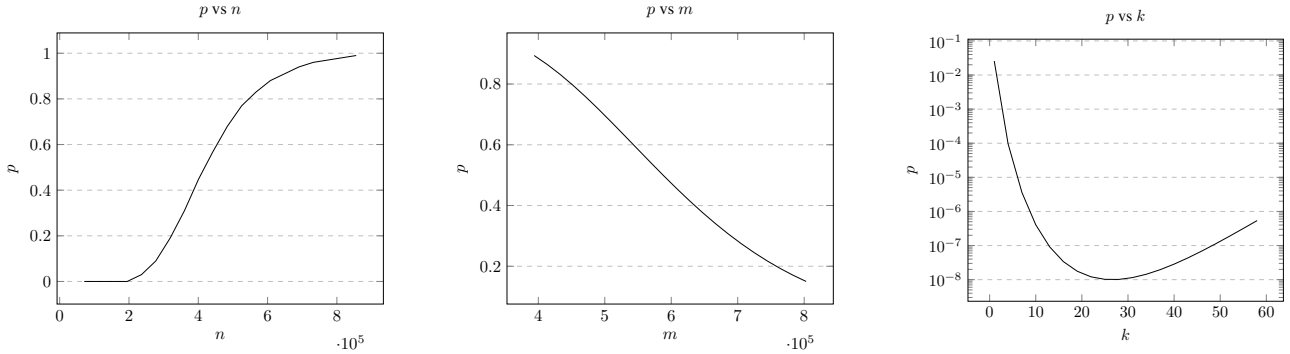


Figure 8: Effect of various filter parameters n, m and k on the false positive probability p.

After Bloom filters are built, their bits are perturbed once more according to RAPPOR (see eq. 18), before being encrypted and sent to client A under a Diffie-Hellman (DH) scheme.

Algorithm 2 Bloom filter constructions for D_A and D_B

Require: $S_C, k_f, \epsilon, \delta, s_k$
  $\mathcal{B}, F \leftarrow \emptyset$
  for $i_i \in S_C$ **do**
    for $L_i \in S_C[i_i]$ **do**
      $H_r \leftarrow$ computeGeoHash(L_i, k_f)
      $\mathcal{B}[L_i]$.append$((H_r, i_i))$
  $\tilde{\mathcal{B}} \leftarrow$ LapNoise$(\mathcal{B}, \epsilon, \delta)$ ▷ See algorithm 3
  for $\mathcal{B}_i \in \tilde{\mathcal{B}}$ **do**
    $F_i \leftarrow$ computeBloomFilter$(\mathcal{B}_i)$
    encrypt(F_i, s_k)
    F.append(F_i)
  return F

The function encrypt encrypts the input from a given private key (DH scheme). We provide the pseudocode for LapNoise below:

Algorithm 3 Laplace noise blocking perturbation

Require: $\mathcal{B}, \epsilon, \delta$
  $\tilde{\mathcal{B}} \leftarrow \emptyset$
  for $\mathcal{B}_i \in \mathcal{B}$ **do**
    $\eta_i \leftarrow$ Lap$(\epsilon, \delta, \Delta\mathcal{B})$
    $\tilde{\mathcal{B}}_i \leftarrow$ add $\eta_i^+ = \max(\eta_i, 0)$ dummy records to $\mathcal{B}_i$
  return $\tilde{\mathcal{B}}$

where $\Delta\mathcal{B}$, the sensitivity of the blocking strategy $\mathcal{B}$ [16] , is calculated by

$$\Delta\mathcal{B} = \max_{D_A} \max_{(D_B, D'_B) \in \mathcal{N}(f_{\bowtie_m}(D_A, \cdot))} \sum_{i=0}^{k} ||\mathcal{B}_i(D_B)| - |\mathcal{B}_i(D'_B)|| \tag{22}$$

4.4 Comparison and classification

The last trivial step of the process is for Alice to decrypt and query Bob's Bloom filters in the set F to identify common locations. While this comparison step does not inherently support APPRL scenarios where exact matching is not sufficient, the pair generation step following the execution of algorithms $\{A_1, \cdots, A_N\}$ can easily be adjusted to support more complex conditions on record features in D_B, after which an additional exchange can be performed with D_A to perform conjunctions and disjunctions on common features. Moreover, this protocol can easily be adapted for the use of ML algorithms as classification primitives, as was shown in section 3.5. Nonetheless, the impact of Bloom filter perturbation steps remains to be quantified in the context of ML linkage accuracy in order to validate such a setup.

4.5 Theoretical analysis

Correctness: Dummy records inserted in algorithm 2 lie in an extended domain and cannot match real entities. Correctness analysis of the pre-processing steps does not deviate from analysis of blocking and pruning strategies, which are widely studied in literature [2]. Overall, the remainder of the protocol does not delete records, hence, mobility linkage achieves the same recall as a non-private blocking protocol.

Privacy: We have that mobility linkage is differentially private:

Theorem 15. *Mobility linkage satisfies* $(\epsilon_A, \epsilon_B, \delta_A, \delta_B, f)$*-OCDP*

[2]False positives can occur in Bloom filters due to their deterministic nature, though we assume their parameters have been set according to the formulas described previously in Section 4.3.

Proof. Alice's view comprises (a) the number of k_f-level bins and (b) the output of Bloom filter construction for each bin. We consider a pair of neighboring datasets $(D_B, D'_B) \in \mathcal{N}(f_{\bowtie_m}(D_A, \cdot))$ differing by one non-matching record only. We have that D_B and D'_B produce bins whose count differs at most by $\Delta\mathcal{B}$ (eq. 22). From lemma B.2 of [16], the Laplace perturbation adds enough records to bound the same number of candidate matching pairs for each $(i, j) \in \mathcal{B}^S$ from D_B and D'_B by e^{ϵ_B}. Therefore, the Laplace perturbation of record bins satisfies (ϵ, δ)-DPRL. For the subsequent Bloom filter construction and comparison step, Alice's view comprises the set of encrypted bins $\tilde{\mathcal{B}}$, including perturbed record counts for each bin. Bloom filter-encoded records for some perturbed bin can only differ by one record in one of the bins, but the output for pair classification will be identical, since they lie in an extended domain and cannot be matched. Bloom filter querying for classification and feature comparison satisfies $(0, \text{negl}(\kappa))$-DPRL, and the maximum number of comparisons is n^2 with $n = \max(|D_A|, |D_B|)$. Hence the Bloom filter construction step and feature exchange satisfy $(0, \text{negl}(\kappa))$-DPRL. A similar proof holds for Alice. By the sequential composition property (if Π_A and Π_B both satisfy OCDP, applying those protocols sequentially satisfies $(\epsilon_A, \epsilon_B, \delta_A, \delta_B, f)$-OCDP), we get the desired result. □

Efficiency: From Theorem 4.8 of [16], we have that given a blocking protocol $\mathcal{B}$ with k bins and a blocking strategy $\mathcal{B}^S$, such that the number of candidate matches for D_A and D_B is $o(n^2)$ with $n = \max(|D_A|, |D_B|)$, if the number of bins k is $o(n^c)$ for $c < 2$ and each bin of a party is compared with $O(1)$ number of bins from the opposite party, the the expected number of candidate matches for Π is sub-quadratic in n. The first condition is guaranteed in mobility linkage by the choice of hash precision level k_f, which determines the end number of bins to be strictly less than the number of records (distance-based blocking). The second condition is satisfied due to the fact that the same precision level is chosen on Alice and Bob's side, yielding one bin comparison per bin hash.

Utility: Mobility linkage isn't compatible with APPRL scenarios out of the box due to the fact that 1) additional features are not encoded inside Bloom filters for immediate comparison in the Bloom filter exchange step and 2) Bloom filters themselves only support set membership queries for fixed identifiers. Nonetheless, support for conjunctions and disjunctions can easily be implemented by adding more subsequent exchange rounds after Bloom filter comparisons. Studies remain to be performed to quantify the impact of such a procedure on privacy. Mobility linkage supports large datasets, moreover, the blocking procedure can be adapted on datasets from different clients with according changes in precision levels if required, as long as the final level k_f is constant across all parties, to match the efficiency requirement given above. Hence, mobility linkage is flexible in the number of additional clients.

5 Experimental results

With the aim of evaluating the performance of PPRL approaches discussed previously, we will implement them in context of a real-world scenario. We start by describing its setting, before benchmarking algorithms with our datasets and discussing obtained results.

5.1 The GUISSTANN project of CREST

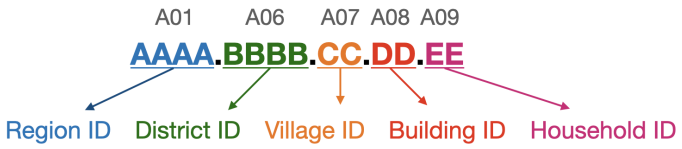
GUISTANN (*GUIder la Statistique publique Sénégalaise, Téléphonie, Algorithmique et Nouvelles techniques Numériques*) was created in an effort to improve public statistics in Senegal using an abundant and inexpensive data source. By training a model on CDR data provided by mobile operators, various demographic indicators can be predicted on a granular geographical level. Such an approach has been successfully implemented in developing countries for establishing poverty maps, improving targeting of populations in need and studying mobility [4] . Our project's contribution include evaluating the prediction results from CDR data thanks to the country's next nationwide census in summer 2023, and enabling prediction of a wide range of health, education, agriculture and mobility indicators.

With this aim, we operate on a reduced population pre-census survey of 10,000 households collected by the ANSD (Agence Nationale de la Statistique et de la Démographie), to be linked with CDR records from Sonatel (the principal telecommunications provider of Senegal). Both datasets are to be kept geographically and operationally isolated due to major data confidentiality concerns. On one side, a government entity would like to contribute to the project with sensitive population records that should not be published for political, cultural and legal reasons. On the other side, a private entity operating in an extremely competitive market would like to contribute sensitive CDR data under the strict condition that competitors and other entities should in no case gain awareness of records or aggregated patterns. We realize that this

setup is adequately fulfilled by the use of a CS VFL training process, as we are working with two feature-partitioned silos with strong privacy requirements

5.1.1 Datasets

The ANSD pre-census dataset includes a range of survey features broken down into the following sections: identification, individual characteristics, deaths, migration, housing, agricultural exploitation and agricultural data. We are interested in performing linkage with CDR data based on unique identification variables, which are structured in the dataset as follows:



ANSD possesses a database of localities with associated coordinates, which allow for geographical identification up to village granularity. Such linkage can be performed locally using trivial methods, since it is performed on exact identifiers. Moreover, precise geolocation of households with GPS-scale precision can also be performed in cases where accuracy is critical, such as in urban areas.

The Sonatel CDR dataset includes automated reports of multiple surveyed events such as incoming and outgoing call metadata, SMS message metadata, mobile data usage and mobile money transfers. Events are typically identified by International Mobile Subscriber Identity (IMSI) numbers, though this information is classified as highly sensitive and cannot even be published for research purposes. Linkage must therefore be performed on the basis of location estimates, which are computed from Base Transmitter Station (BTS) identifiers. Much like in ANSD's case, mobile operators such as Sonatel keep track of a database of cell towers, which can be used to associate IDs with geographical coordinates to obtain an approximate location of the user. The precision of this estimate can vary from 20m of accuracy with latest 5G antennas to 15km in rural LTE deployments. We hence let operators provide these estimates using internal linkage on exact identifiers.

Our resulting record linkage setup can hence be summarized as follows:

Table 4: D_A: ANSD census data

Household location	Individual characteristics	...
(lat/lng)	(survey answers)	...
...	...	...

Table 5: D_B: Sonatel CDR data

BTS Location	Call duration	...
(lat/lng)	(duration)	...
...	...	...

The construction of D_A and D_B on which geographical linkage is performed from raw ANSD and Sonatel data can be summarized as follows:

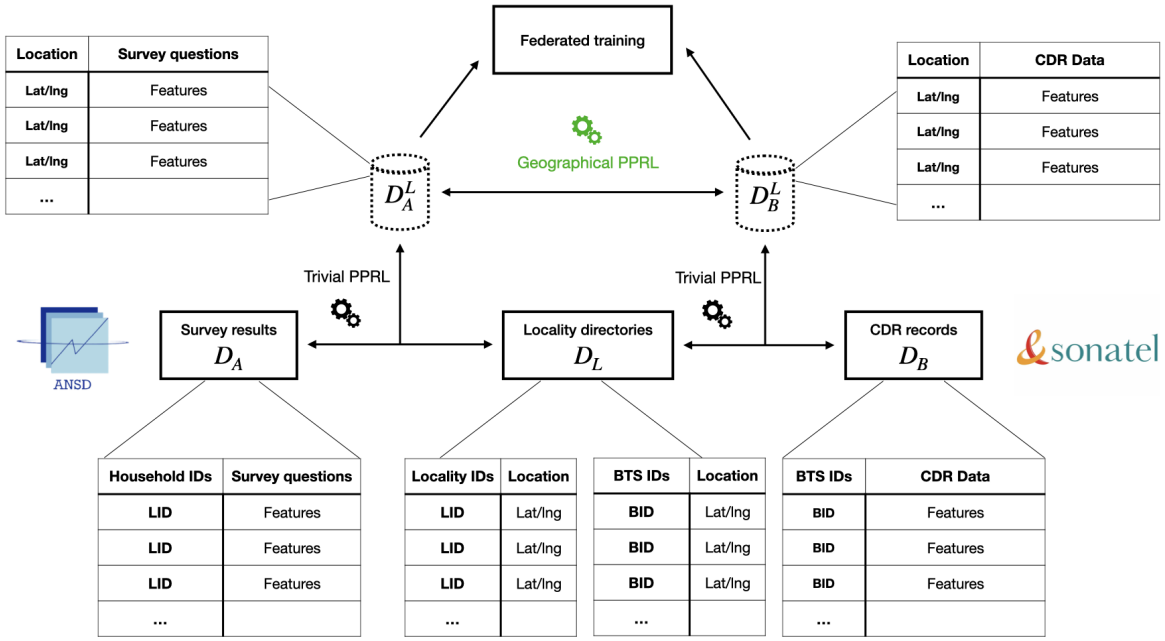


Figure 9: GUISSTANN dataset structure with records identified by location IDs for survey data, and CDR records identified by cell tower IDs (BTS IDs) for mobile data. Both location IDs and BTS IDs are resolved locally into geographical coordinates using direct linkage methods. Survey records are then linked to CDR entries using geographical PPRL.

5.1.2 Pre-processing

The Sonatel CDR dataset comprises 90,000,000 unique records for some 180,000 individuals. The setup summarized in table 5 presents the additional challenge of inferring household locations from mobility patterns. Since common identifiers such as International Mobile Subscriber Identity (IMSI) have been removed from datasets, we are unable to link surveyed home locations with CDR identities, or run supervised learning algorithms on ground truth data. We hence apply the pre-processing step described in Section 4.1 to infer a set of potential home locations for every entity's CDR record set. Existing studies [2] combined with analysis of records in the case of Senegal allow us to identify five criteria for defining likely household locations:

1. The place where the majority of incoming and outgoing calls and texts are made, within the 7pm-6am range. Clusters outside this range have been found to be easily misclassified from work locations of individuals.
2. The place with the maximal number of distinct days for which traffic is recorded, including weekends.
3. The place where all recorded activities can be aggregated within a 1000 meter range.
4. Where applicable, the place where the maximal number of mobile money transactions have been recorded on the 7pm-6am range.
5. When available, the place where the most consecutive activity points are recorded in proximity of other household members. This requires knowledge of at least 1 close contact of the person of interest and provides additional challenges.

We run several combinations of the five criteria described above on the Sonatel CDR dataset and collect locations $L_1, \ldots, L_n$, where L_i is the actual detected home location by a particular algorithm i.

Example 16. Consider the following simplified CDR records for a certain individual:

User	Type	Cell ID	Timestamp	...
38409	outgoing_call	24	1643057282	
38409	outgoing_call	23	1643081266	
38409	incoming_call	24	1643237354	
38409	outgoing_call	24	1643237483	
38409	incoming_call	53	1643259592	
38409	incoming_call	23	1643664724	

(a) Sample CDR records

Cell ID	Coordinates
23	14.7054642,-17.4541088
24	14.7054337,-17.4553528
53	14.7081035,-17.4534987

(b) Sample cell tower locations

We apply algorithm A_1, which considers only criteria #1. Based on this criteria, we have that L_1 is at the location of cell 24. The according spatial uncertainty for L_1 becomes

$$\text{SU}_{L_1} = \frac{2 \cdot 140}{3 \cdot 2} + \frac{1 \cdot 350}{3 \cdot 2} = 105 \text{ m}$$

5.2 Evaluation

We evaluate surveyed protocols, as well as our mobility linkage approach on the Sonatel and ANSD datasets described previously. Using a prototype implementation of linkage algorithms in Python (version `3.8.10`), experiments were run on a laptop with an `Apple M1 Max` (3.2 GHz) SOC with 64 GB of main memory and macOS Ventura 13.2.1. We use two different setups to evaluate the flexibility of algorithms:

1. A simple EPPRL scenario whereby comparisons are solely based on GeoHash distances.
2. A more involved APPRL case for which a complex matching rule involving conjunctions and disjunctions is used for comparisons:

$$I_t^m(r,s) = \begin{cases} 1 & (06:00 \leq r.\text{time} \leq 19:00 \wedge m(r.\text{loc}, s.\text{loc}) < 2t) \vee \\ & \quad (19:00 \leq r.\text{time} \leq 06:00 \wedge m(r.\text{loc}, s.\text{loc}) < t) \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

We first evaluate the naive linkage method (APC) with $c = |D_A| \times |D_B|$ comparisons as a baseline. We also evaluate the performance gains induced by pruning, which is performed on the basis of a simple distance cutoff as pre-processing, which, while effective in slashing the size of the comparison pool (up to 99.11% of savings), leads to a slight decrease in recall due to the sparsity of cells in rural areas. The cutoff threshold can be adjusted to include the greatest distance in the set of true positives, though at the expense of efficiency savings.

We then evaluate the one-sided DH-PSI method with blocking based on the administrative regions of locations in both datasets. We notice large performance gains due to the size reduction of the comparison pool, and we notice that much like pruning, most of the computational load is offset to the pre-processing phase. In this testing case scenario, the simple exact matching rule is sufficient to achieve an optimal recall though in more complete settings, blocking is likely to have a negative impact on this metric.

Mobility Linkage: Finally, we evaluate the performance of our method described in section 4. Parameters are chosen as follows:

- GeoHash precision levels: $k_c = 3$ for the primary hash (initial record pruning), $\Delta_K = 1$ to reach $k = 4$ for key cell computations, and $k_f = 5$ for bloom filter computations, to reach a final precision of ± 2.4km per bin.
- Algorithms: A_1, A_2 and A_3 based on home location criteria #1, #2 and #3, returning top 2 candidates each.
- Thresholds: c_t such that the distance between and the centroid exceeds 1/10 of the cell's edge width, which, working with GeoHash level 4, corresponds to a distance of 9 km. $S_{max} = 0.9$
- Laplace coefficients: $\epsilon = 1.6$, $\delta = 10^{-5}$ based on previous studies, maximizing recall efficiency while keeping strong privacy guarantees.

When inserting dummy records for the Laplace noise perturbation step, the extended domain is defined on the temporal plane, for which records are then verified by Alice during classification.

Linkage algorithm	Subscribers	Records	c	Pre-process time (s)	Link time (s)	Total time (s)	Recall
Naive linkage	100	1000	10^5	0.106	1.86	1.966	1
Naive linkage	100	10000	10^6	0.701	19.353	20.054	1
Naive linkage	100	100000	10^7	6.55	194.113	200.663	1
Naive linkage	1000	10000	10^7	0.753	194.069	194.822	1
Naive linkage + pruning	1000	10000	10^4	190.886	0.371	191.257	0.98
Naive linkage	1000	100000	10^8	6.730	1996.512	2003.242	1
Blocking + PSI	100	1000	10^5	0.102	0.002	0.104	1
Blocking + PSI	100	10000	10^6	0.852	0.002	0.854	1
Blocking + PSI	100	100000	10^7	6.295	0.005	6.300	1
Blocking + PSI	1000	100000	10^8	12.720	0.015	12.735	1
Blocking + PSI	1000	1000000	10^9	82.329	0.154	82.483	1
Blocking + PSI	10000	1000000	10^{10}	243.86	0.162	244.022	1
Mobility Linkage	100	1000	10^5	1.412	0.344	1.756	0.99
Mobility Linkage	100	10000	10^6	2.878	0.594	3.472	0.99
Mobility Linkage	1000	10000	10^7	4.851	0.829	5.680	0.99
Mobility Linkage	1000	100000	10^8	7.177	2.142	9.319	0.99
Mobility Linkage	10000	100000	10^9	11.630	3.927	15.557	0.99
Mobility Linkage	10000	1000000	10^{10}	20.963	5.251	26.214	0.99
Mobility Linkage	10000	10000000	10^{11}	52.583	6.293	59.876	0.99

Table 7: Summary of protocol executions for different linkage algorithms in the EPPRL scenario. Values for each entry are obtained by averaging 10 repeats. c denotes the total number of comparisons.

We notice that mobility linkage achieves satisfactory performance for large datasets, though at an increased baseline pre-processing cost due to pruning, blocking and Bloom filter construction. Here, a trivial optimization for the computation of GeoHashes is to immediately compute k_f and truncate the output accordingly to obtain $k_c + \Delta_k$ and k_c. The decrease in recall is due to outliers in the set of CDR records that do not completely satisfy criteria #1, #2 and #3, though algorithms can be adjusted accordingly in further studies.

Linkage algorithm	Subscribers	Records	c	Pre-process time (s)	Link time (s)	Total time (s)	Recall
Naive linkage	100	1000	10^5	0.091	2.187	2.278	1
Naive linkage	100	10000	10^6	0.661	21.810	22.471	1
Naive linkage	100	100000	10^7	6.364	216.765	223.129	1
Naive linkage	1000	10000	10^7	0.726	212.470	213.196	1
Naive linkage + pruning	1000	10000	10^7	191.644	0.424	192.068	0.98
Naive linkage	1000	100000	10^8	6.630	2122.370	2129.000	1

Table 8: Summary of protocol executions for different linkage algorithms in the APPRL scenario. Values for each entry are obtained by averaging 10 repeats.

For the APPRL scenario, we execute our naive approach on the conjugate rule 23, for which we notice increased execution times, as expected. Due to the fact that PSI protocols do not natively support such rules, and that expansion is not an option for our target number of comparisons, we survey an approach involving an unsupervised k-means classifier instead.

As mentioned previously, the base mobility linkage protocol does not support conjunction and disjunction rules out of the box, though this can be remediated by adding additional rounds of comparison subsequently to the Bloom filter query phase. It remains to quantify the performance of such an approach.

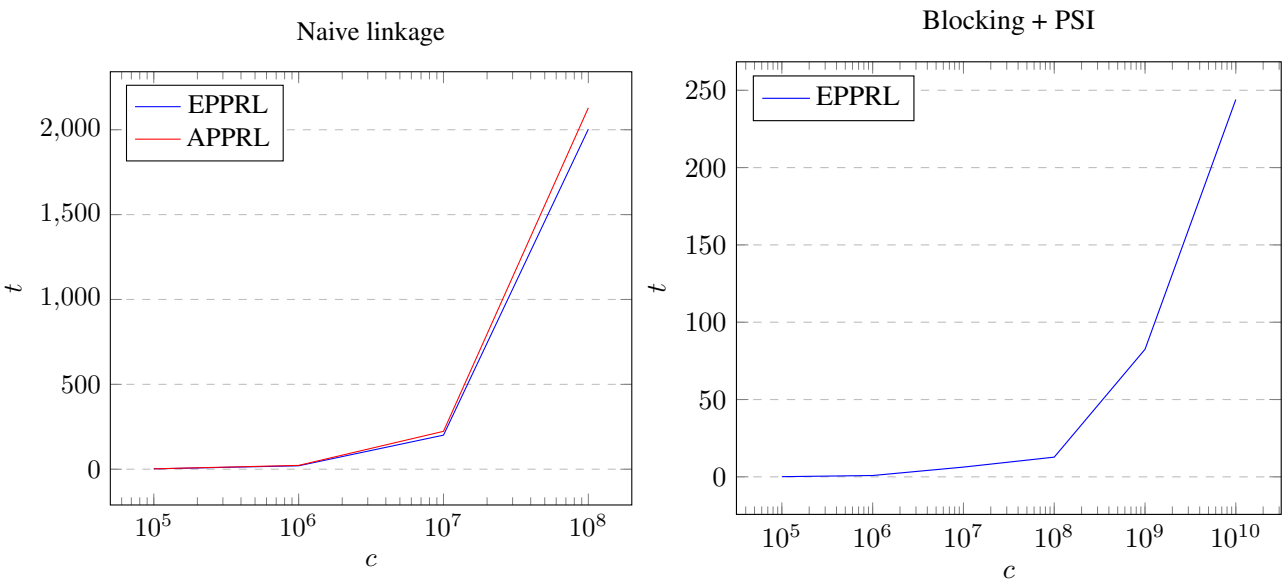


Figure 10: Execution time of the surveyed protocols vs. total number of comparisons c

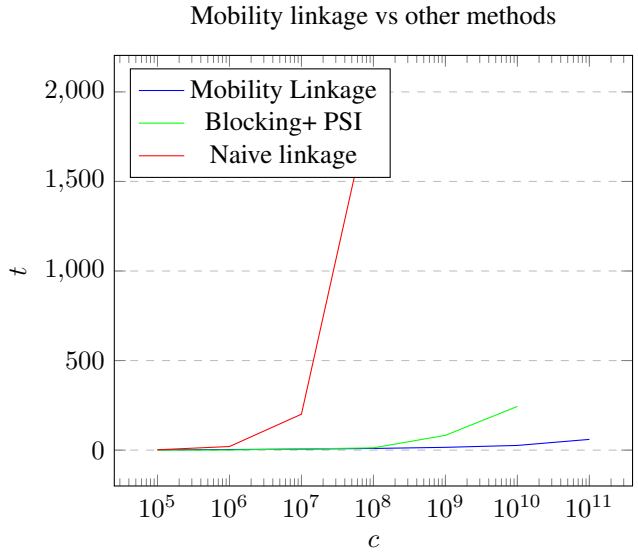


Figure 11: Execution time of mobility linkage vs. total number of comparisons c

6 Conclusion

Privacy-Preserving Record Linkage (PPRL) is a key area of research for the collaboration of different organizations on Machine Learning (ML) models, without compromising the privacy and confidentiality of entities in private datasets. Cross-Silo Federated Learning (CSFL) frameworks constitute an operational solution to the physical isolation of datasets in the context of model training, though at the cost of additional privacy and efficiency challenges associated with Big Data. In this paper, we provided background material required to evaluate the setting, process and challenges of PPRL in context of a real-world scenario, and we surveyed existing PPRL approaches against a common framework. The proposed protocol for mobility linkage addresses common efficiency and privacy pitfalls associated with PPRL on PII data, while achieving satisfactory accuracy results. A limitation to this approach is that approximate/fuzzy matching rules are not directly supported by the Bloom filter comparison primitive, though we can easily propose extensions involving a classifi-

cation step using supervised/unsupervised learning algorithm, adding an additional secure exchange step between parties for needed features. Our protocol directly supports scenarios whereby a third party must perform the pair classification, and is scalable enough to function with additional clients.

6.1 Further work

The accuracy analysis of surveyed protocols would benefit from specifically evaluating their impact on Cross-Silo Federated Learning pipelines. In particular, surveying linkage mistakes in such contexts would give valuable insights on how to enhance the effectiveness of PPRL as a pre-processing step to FL. In the context of computer vision, support for efficient linkage of additional modalities such as image and sound data could benefit a range of applications. One can envision for instance satellite imagery-enabled geographical linkage for population surveys, in which this particular data source can bring additional information to the matching process. More research is also needed to study remediation to membership inference attacks within the context of FL pipelines; data such as CDR records are straightforward to craft maliciously in an attempt to be matched with real records to infer information about private datasets.

Acknowledgements I would like to extend my gratitude to Professor Guillaume Hollard from CREST for leading and welcoming me into the outstanding GUISSTANN project, which paves the way for further impactful machine learning projects on Senegalese demographics. I am equally grateful to Idrissa Diagne for his warm welcome and hospitality at ENSAE in Dakar, as well as Director Oumar Fall from ANSD for his continued support throughout our visit of the institute. Moustapha Mbengue, Bourama Mane, Anta Gueye and Alioune Tine provided helpful discussions and insights on various aspects of the 2023 population census at the core of our project. I would also like to mention Richard Ayizou, Hamady Diallo, Lucia Carai and Alejandro Christlieb, with whom I have the pleasure to work with in Senegal. Lastly, I am grateful to Professor Jesse Read from LIX for his feedback throughout my internship.

7 References

[1] Rohan Baxter, Peter Christen, and Centre Epidemiology. A Comparison of Fast Blocking Methods for Record Linkage. *Proc. of ACM SIGKDD'03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, July 2003.

[2] Iva Bojic, Emanuele Massaro, Alexander Belyi, Stanislav Sobolevsky, and Carlo Ratti. *Choosing the Right Home Location Definition Method for the Given Dataset*, volume 9471. October 2015.

[3] Feng Chen, Xiaoqian Jiang, Shuang Wang, Lisa M. Schilling, Daniella Meeker, Toan Ong, Michael E. Matheny, Jason N. Doctor, Lucila Ohno-Machado, and Jaideep Vaidya. Perfectly Secure and Efficient Two-Party Electronic-Health-Record Linkage. *IEEE Internet Computing*, 22(2):32–41, March 2018.

[4] Guanghua Chi, Han Fang, Sourav Chatterjee, and Joshua E. Blumenstock. Micro-Estimates of Wealth for all Low- and Middle-Income Countries. *Proceedings of the National Academy of Sciences*, 119(3):e2113658119, January 2022. arXiv:2104.07761 [cs, econ, q-fin].

[5] Peter Christen. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 24, January 2011.

[6] Peter Christen. Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection / peter christen. 2012.

[7] Peter Christen and Karl Goiser. Quality and Complexity Measures for Data Linkage and Deduplication. In Fabrice J. Guillet and Howard J. Hamilton, editors, *Quality Measures in Data Mining*, pages 127–151. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[8] David Clayton, Christopher Patton, and Thomas Shrimpton. Probabilistic Data Structures in Adversarial Environments. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1317–1334, London United Kingdom, November 2019. ACM.

[9] Aiden Durrant, Milan Markovic, David Matthews, David May, Jessica Enright, and Georgios Leontidis. The role of cross-silo federated learning in facilitating data sharing in the agri-food sector. *Computers and Electronics in Agriculture*, 193:106648, February 2022.

[10] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067, November 2014. arXiv:1407.6981 [cs].

[11] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In Christian Cachin and Jan L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 1–19, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[12] Luca Gagliardelli, George Papadakis, Giovanni Simonini, Sonia Bergamaschi, and Themis Palpanas. Generalized Supervised Meta-blocking (technical report), April 2022. arXiv:2204.08801 [cs].

[13] Zakaria Gheid and Yacine Challal. Private and Efficient Set Intersection Protocol for Big Data Analytics. In Shadi Ibrahim, Kim-Kwang Raymond Choo, Zheng Yan, and Witold Pedrycz, editors, *Algorithms and Architectures for Parallel Processing*, pages 149–164, Cham, 2017. Springer International Publishing.

[14] Rob Hall and Stephen Fienberg. *Privacy-Preserving Record Linkage*. January 1970. Pages: 283.

[15] Shumin Han, Derong Shen, Tiezheng Nie, Yue Kou, and Ge Yu. Private Blocking Technique for Multi-party Privacy-Preserving Record Linkage. *Data Science and Engineering*, 2(2):187–196, June 2017.

[16] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. Composing Differential Privacy and Secure Computation: A case study on scaling private record linkage, September 2017. arXiv:1702.00535 [cs].

[17] Mikko A. Heikkilä, Antti Koskela, Kana Shimizu, Samuel Kaski, and Antti Honkela. Differentially private cross-silo federated learning, July 2020. arXiv:2007.05553 [cs, stat].

[18] Chao Huang, Jianwei Huang, and Xin Liu. Cross-Silo Federated Learning: Challenges and Opportunities, June 2022. arXiv:2206.12949 [cs].

[19] Ali Inan, Murat Kantarcioglu, Gabriel Ghinita, and Elisa Bertino. Private record matching using differential privacy. *Cyber Center Publications*, March 2010.

[20] J. Cao, F. -Y. Rao, E. Bertino, and M. Kantarcioglu. A hybrid private record linkage scheme: Separating differentially private synopses from matching records. In *2015 IEEE 31st International Conference on Data Engineering*, pages 1011–1022, April 2015. Journal Abbreviation: 2015 IEEE 31st International Conference on Data Engineering.

[21] J. Mori, I. Teranishi, and R. Furukawa. Continual Horizontal Federated Learning for Heterogeneous Data. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2022. Journal Abbreviation: 2022 International Joint Conference on Neural Networks (IJCNN).

[22] Dimitrios Karapiperis and Vassilios S. Verykios. A fast and efficient Hamming LSH-based scheme for accurate linkage. *Knowledge and Information Systems*, 49(3):861–884, December 2016.

[23] Shahidul Islam Khan, Abir Bin Ayub Khan, and Abu Sayed Md Latiful Hoque. Privacy preserved incremental record linkage. *Journal of Big Data*, 9(1):105, November 2022.

[24] Abel N Kho, John P Cashy, Kathryn L Jackson, Adam R Pah, Satyender Goel, Jörn Boehnke, John Eric Humphries, Scott Duke Kominers, Bala N Hota, Shannon A Sims, Bradley A Malin, Dustin D French, Theresa L Walunas, David O Meltzer, Erin O Kaleba, Roderick C Jones, and William L Galanter. Design and implementation of a privacy preserving electronic health record linkage tool in Chicago. *Journal of the American Medical Informatics Association*, 22(5):1072–1080, September 2015.

[25] Mehmet Kuzu, Murat Kantarcioglu, Ali Inan, Elisa Bertino, Elizabeth Durham, and Bradley Malin. Efficient Privacy-Aware Record Integration. *Advances in database technology : proceedings. International Conference on Extending Database Technology*, pages 167–178, March 2013.

[26] Pierre K. Y. Lai, Siu-Ming Yiu, K. P. Chow, C. F. Chong, and Lucas Chi Kwong Hui. An Efficient Bloom Filter Based Solution for Multiparty Private Matching. In *Security and Management*, 2006.

[27] Ali Lawati, Dongwon Lee, and Patrick McDaniel. *Blocking-aware private record linkage*. June 2005. Pages: 68.

[28] Chen Li, Liang Jin, and Sharad Mehrotra. Supporting Efficient Record Linkage for Large Data Sets Using Mapping Techniques. *World Wide Web*, 9:557–584, December 2006.

[29] Fengjun Li, Yuxin Chen, Bo Luo, Dongwon Lee, and Peng Liu. Privacy Preserving Group Linkage. In Judith Bayard Cushing, James French, and Shawn Bowers, editors, *Scientific and Statistical Database Management*, pages 432–450, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[30] Katrina Ligett, Seth Neel, Aaron Roth, Bo Waggoner, and Z. Steven Wu. Accuracy First: Selecting a Differential Privacy Level for Accuracy-Constrained ERM, May 2017. arXiv:1705.10829 [cs].

[31] Yang Liu, Yan Kang, Tianyuan Zou, Yanhong Pu, Yuanqin He, Xiaozhou Ye, Ye Ouyang, Ya-Qin Zhang, and Qiang Yang. Vertical Federated Learning, November 2022. arXiv:2211.12814 [cs].

[32] Mohamed Maouche, Sonia Mokhtar, and Sara Bouchenak. HMC: Robust Privacy Protection of Mobility Data against Multiple Re-Identification Attacks. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2:1–25, September 2018.

[33] Christine O'Keefe, Ming Yung, Lifang Gu, and Rohan Baxter. *Privacy preserving data linkage protocols*. October 2004. Pages: 102.

[34] Pascal Paillier. *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, volume 5. May 1999. Journal Abbreviation: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes Pages: 238 Publication Title: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes.

[35] Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, Sudeep Agarwal, Julien Freudiger, Andrew Byde, Abhishek Bhowmick, Gaurav Kapoor, Si Beaumont, Áine Cahill, Dominic Hughes, Omid Javidbakht, Fei Dong, Rehan Rishi, and Stanley Hung. Federated evaluation and tuning for on-device personalization: System design applications, 2022.

[36] Benny Pinkas, Thomas Schneider, and Michael Zohner. Faster private set intersection based on OT extension. *ACM Transactions on Privacy and Security*, 21:797–812, January 2018.

[37] Thilina Ranbaduge, Peter Christen, and Dinusha Vatsalan. *Tree based scalable indexing for multi-party privacy-preserving record linkage*. November 2014.

[38] Thilina Ranbaduge, Dinusha Vatsalan, and Peter Christen. *Clustering-Based Scalable Indexing for Multi-party Privacy-Preserving Record Linkage*. May 2015.

[39] Fang-Yu Rao, Jianneng Cao, Elisa Bertino, and Murat Kantarcioglu. Hybrid Private Record Linkage: Separating Differentially Private Synopses from Matching Records. *ACM Transactions on Privacy and Security*, 22(3):1–36, August 2019.

[40] Mike Rosulek and Ni Trieu. Compact and malicious private set intersection for small sets. Cryptology ePrint Archive, Paper 2021/1159, 2021. https://eprint.iacr.org/2021/1159.

[41] Rainer Schnell, Tobias Bachteler, and Jörg Reiher. A novel error-tolerant anonymous linking code. 2011.

[42] Rainer Schnell, Tobias Bachteler, and Jörg Reiher. Privacy-preserving record linkage using Bloom filters. *BMC medical informatics and decision making*, 9:41, September 2009.

[43] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks against Machine Learning Models, March 2017. arXiv:1610.05820 [cs, stat].

[44] Rebecca C. Steorts, Samuel L. Ventura, Mauricio Sadinle, and Stephen E. Fienberg. A Comparison of Blocking Methods for Record Linkage, July 2014. arXiv:1407.3191 [cs, stat].

[45] Maarten Vanhoof, Fernando Reis, Zbigniew Smoreda, and Thomas Ploetz. Detecting home locations from CDR data: introducing spatial uncertainty to the state-of-the-art, August 2018. arXiv:1808.06398 [cs].

[46] Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Information Systems*, 38(6):946–969, September 2013.

[47] W. Xue, D. Vatsalan, W. Hu, and A. Seneviratne. Sequence Data Matching and Beyond: New Privacy-Preserving Primitives Based on Bloom Filters. *IEEE Transactions on Information Forensics and Security*, 15:2973–2987, 2020.

[48] Kang Wei, Jun Li, Chuan Ma, Ming Ding, Sha Wei, Fan Wu, Guihai Chen, and Thilina Ranbaduge. Vertical Federated Learning: Challenges, Methodologies and Experiments, February 2022. arXiv:2202.04309 [cs].

[49] Kevin Witlox. *Secure blocking for record linkage*. PhD thesis, University of Twente, Overjissel, The Netherlands.

[50] Zhaomin Wu, Qinbin Li, and Bingsheng He. A Coupled Design of Exploiting Record Similarity for Practical Vertical Federated Learning, February 2023. arXiv:2106.06312 [cs].

[51] Qingsong Ye, Ron Steinfeld, Josef Pieprzyk, and Huaxiong Wang. Efficient Fuzzy Matching and Intersection on Private Datasets. In Donghoon Lee and Seokhie Hong, editors, *Information, Security and Cryptology – ICISC 2009*, pages 211–228, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

A Appendix

A.1 List of employed acronyms

ANSD: Agence Nationale de la Statistique et de la Démographie
APC: All Pairwise Comparisons
APPRL: Approximative Privacy-Preserving Record Linkage
BTB: Bit Tree Blocking
BTS: Base Transceiver Station
CDR: Call Detail Record
DH: Diffie-Hellmann
DP: Differential Privacy
EPPRL: Exact Privacy-Preserving Record Linkage
ER: Entity Resolution
FL: Federated Learning
FPS: Frequent Pairs Schemes
IND-S2PC: Indistinguishability of secure two-party computations
IMSI: International Mobile Subscriber Identity
IRL: Incremental Record Linkage
LP: Laplace Protocol
LPH: Locality Preserving Hash
LSH: Locality Sensitive Hash
ML: Machine Learning
OCDP: Output-Constrained Differential Privacy
PII: Personally Identifiable Information
PRL: Private Record Linkage
PPRL: Privacy-Preserving Record Linkage
PSI: Private Set Intersection
QID: Quasi-IDentifiers
RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response
RL: Record Linkage
SLK: Statistical Linkage Keys
SMC: Simple Matching Coefficients
SK: Secret Key
SMC: Secure Multiparty Computation
VFL: Vertical Federated Learning

k	lat error	lng error	km error
1	± 23	± 23	± 2500
2	± 2.8	± 5.6	± 630
3	± 0.70	± 0.70	± 78
4	± 0.087	± 0.18	± 20
5	± 0.022	± 0.022	± 2.4
6	± 0.0027	± 0.0055	± 0.61
7	± 0.00068	± 0.00068	± 0.076
8	± 0.000085	± 0.00017	± 0.019

Table 9: Geohash encoding length k vs lat/lng error and precision in km

A.2 Supporting definitions

Definition 17 (Personally Identifiable Information (PII)). Any information about an individual maintained by an agency, including (1) any information that can be used to distinguish or trace an individual‘s identity, such as name, social

security number, date and place of birth, mother's maiden name, or biometric records; and (2) any other information that is linked or linkable to an individual, such as medical, educational, financial, and employment information.

Definition 18 (**Decisional DH assumption**)**.** For a multiplicative cyclic group G, a generator g and integers $a, b, c \in \mathbb{Z}$, the decisional DH problem is hard, if for every probabilistic polynomial adversary $\mathcal{A}$,

$$|\Pr[A(g, g^a, g^b, g^{ab}) = 1] - \Pr[A(g, g^a, g^b, g^c) = 1]| < \epsilon \tag{24}$$

where the probability is taken over (g, a, b, c).

Definition 19 (Lap$(\epsilon, \delta, \Delta\mathcal{B})$ distribution)**.** A random variable follows the Lap$(\epsilon, \delta, \Delta\mathcal{B})$ distribution if it has a probability density function

$$\Pr[n = x] = p \cdot \exp(-(\epsilon/\Delta\mathcal{B})|x - \eta^0|), \quad \forall x \in \mathbb{Z} \tag{25}$$

with $p = \frac{\exp(\epsilon/\Delta\mathcal{B})-1}{\exp(\epsilon/\Delta\mathcal{B})+1}$ and $\eta^0 = -\frac{\Delta\mathcal{B}\ln((\exp(\epsilon/\Delta\mathcal{B})+1)(1-(1-\delta)^{1/\Delta\mathcal{B}}))}{\epsilon}$