Federated Binary Matrix Factorization Using Proximal Optimization

Sebastian Dalleiger,[1] Jilles Vreeken,[2] Michael Kamp[3]

[1]KTH Royal Institute of Technology
[2]CISPA Helmholtz Center for Information Security
[3]Institute for AI in Medicine, UK Essen and Ruhr University Bochum, Monash University
sdall@kth.se, jv@cispa.de, michael.kamp@uk-essen.de

Abstract

Identifying informative components in binary data is an essential task in many application areas, including life sciences, social sciences, and recommendation systems. Boolean matrix factorization (BMF) is a family of methods that performs this task by factorizing the data into dense factor matrices. In real-world settings, the data is often distributed across stakeholders and required to stay private, prohibiting the straightforward application of BMF. To adapt BMF to this context, we approach the problem from a federated-learning perspective, building on a state-of-the-art continuous binary matrix factorization relaxation to BMF that enables efficient gradient-based optimization. Our approach only needs to share the relaxed component matrices, which are aggregated centrally using a proximal operator that regularizes for binary outcomes. We show the convergence of our federated proximal gradient descent algorithm and provide differential privacy guarantees. Our extensive empirical evaluation shows that our algorithm outperforms, in quality and efficacy, federation schemes of state-of-the-art BMF methods on a diverse set of real-world and synthetic data.

Replication Material — doi.org/10.5281/zenodo.14501661

1 Introduction

Discovering patterns and dependencies in distributed binary data sources is a common problem in many applications, such as cancer genomics (Liang, Zhu, and Lu 2020), recommender systems (Ignatov et al. 2014), and neuroscience (Haddad et al. 2018). Data is often distributed horizontally (i.e., the rows of the data matrix are split across hosts) and may not be pooled. For example, biopsies are performed in different hospitals, with each location measuing the expression of a common set of genes. Although there exists an explicit interest in analyzing this data jointly, privacy regulations mandate that these measurements may not be shared, thereby limiting the applicability of traditional centralized methods.

Federated learning (McMahan et al. 2017) enables learning from distributed datasets *without* disclosing sensitive data. Existing methods for federated non-negative matrix factorization (Li et al. 2021) are specific to real-valued data, and similar to non-federated Non-negative Matrix Factorization (NMF) (Paatero and Tapper 1994; Lee and Seung 1999, 2000), singular value decomposition (Golub and Loan 1996), and principal component analysis (Golub and Loan 1996), do not achieve interpretable results for binary data (Miettinen et al. 2008; Dalleiger and Vreeken 2022).

Boolean Matrix Factorization (BMF) alleviates this problem by approximating a *centralized* Boolean target matrix $A \in \{0,1\}^{n \times m}$ by the Boolean product

$$A \approx [U \circ V]_{ij} = \bigvee_{l \in [k]} U_{il} \wedge V_{lj}$$

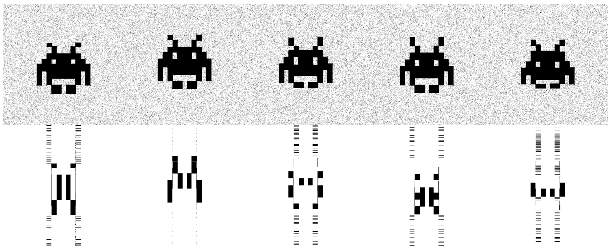
of two low-rank Boolean factor matrices (Miettinen et al. 2008), $U \in \{0,1\}^{n \times k}$ (*feature matrix*) and $V \in \{0,1\}^{k \times m}$ (*coefficient matrix*).

Although there are myriad heuristics to approximate this NP-hard problem, doing so for *distributed data* without sharing private information remains an open problem. Even though we could approach distributed binary data with standard federated learning techniques, e.g. aggregating locally-obtained BMF results into a shared matrix, this requires an aggregation into binary values, such as *rounded average*, *majority vote*, and *logical* `or`. Such techniques, however, lack the precision required by binary data.

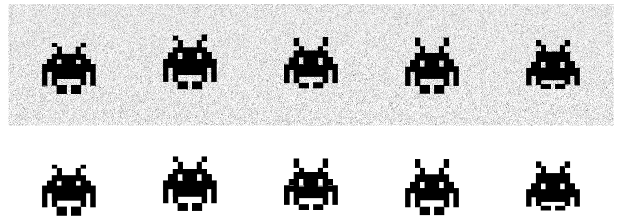
To visualize the extent of this problem, we show the impact of straightforward aggregation in Fig. 1(a), which highlights that even the best combination of a local factorization algorithm and an aggregation scheme—here, Asso (Miettinen et al. 2008) using logical `or`—leads to bad reconstructions.

Recently, Dalleiger and Vreeken (2022) showed we can continuously relax BMF into a regularized *binary* matrix factorization problem using linear (rather than Boolean) algebra and proximal gradients, yielding an efficient and scalable approach with state-of-the-art performance. Taking advantage of this relaxation, we propose the Felb algorithm that locally factorizes while centrally, yet privacy-consciously aggregates coefficients using a proximal aggregation, thereby efficiently yielding valid global binary matrices. On our toy example in Fig. 1(b) it achieves a nearly perfect reconstruction.

We show that Felb converges to binary matrices, provide differential privacy guarantees using, e.g., the Gaussian mechanism (Balle and Wang 2018), and we experimentally validate that the utility remains high. Moreover, we demonstrate that Felb outperforms baselines derived via straightforward parallelization of state-of-the-art BMF methods on numerous real-world and synthetic datasets.



(a) Aggregated BMF



(b) Our Method (FELB)

Figure 1: Our method reconstructs distributed Boolean data well. Representing 1s as black pixels, for (a) ELBMF using logical `or` aggregation and (b) our novel federated factorization called FELB, we show (top row) the client-data subjected to additive noise, and (bottom row) the aggregation-based reconstructions.

In summary, our main contributions are as follows:

- We introduce a novel federated proximal-gradient-descent for BMF (FELB).
- We improve over the state-of-the-art in BMF with our adaptive regularization (FELB$^{\text{MU}}$).
- Provide a formal foundation for federated inertial alternating proximal-gradient optimization under non-convex regularization.
- We experimentally show that our methods are both efficient and accurate.

2 Related Work

To the best of our knowledge, there exists no federated BMF algorithms. We therefore primarily discuss the relations to *BMF*, and *federated factorization*, and *federated learning*.

We distinguish two classes of **BMF** methods: First, *discrete optimization-based methods* that use Boolean algebra, such as ASSO (Miettinen et al. 2008) using a set-cover-like approach, GRECOND (Belohlávek and Vychodil 2010), MEBF (Wan et al. 2020) using fast geometric segmentation, or SOFA (Neumann and Miettinen 2020) based on streaming clustering. Second, *continuous optimization-based methods* that use linear algebra for solving the binary matrix factorization problem, introduced by Zhang et al. (2007), and advanced by Araujo, Ribeiro, and Faloutsos (2016) based on thresholding, and by Hess et. al (Hess, Morik, and Piatkowski 2017; Hess and Morik 2017) using a proximal operator. Combining ideas from the two complementary regularization strategies of Hess, Morik, and Piatkowski (2017) and Zhang et al. (2007), Dalleiger and Vreeken (2022) recently removed the need for post-processing via a proximal operator for an elastic-net-based regularizer.

With regards to **federated factorization** in general, 'parallel' algorithms for matrix factorization (Yu et al. 2014) as well as binary matrix factorization (Khanna et al. 2013) seek computational efficiency without addressing privacy concerns. Towards matrix factorization for distributed privacy-sensitive data, methods exist for federated matrix factorization (Du et al. 2021) and federated non-negative matrix factorization (Li et al. 2021). These methods, however, are not specialized to Boolean matrices. Here, we close the research gap by addressing the need for a federated, privacy-preserving binary (or Boolean) matrix factorization algorithm.

Recent advances in **federated learning** involve techniques like FedProx (Li et al. 2020a) and SCAFFOLD (Karimireddy et al. 2020). FedProx, an extension of FedAvg (McMahan et al. 2017), introduces a proximity penalty term to stabilizing the training process across different clients. SCAFFOLD enhances federated learning by correcting client drift using variance reduction techniques, thereby improving convergence rates and model accuracy compared to traditional methods like FedAvg, while ProxSkip (Mishchenko et al. 2022) uses randomization to reduce the computational cost of proximal operators which are significantly more expensive than our operators. Despite these advances, most research focuses on training deep neural networks using stochastic-gradient-based local optimization schemes. These approaches often yield to a slow convergence to suboptimal non-Boolean solutions, if they are deployed to similar non-convex alternating optimization contexts.

3 Federated Proximal Binary Matrix Factorization

Having contextualized our problem, we now formally introduce our federated Boolean matrix factorization scenario, show how we separate our problem into manageable subproblems; describe how to efficiently and solve subproblems in terms of binary matrix factorization relaxation, while preserving privacy; and formally show that we compute a Boolean matrix factorization upon convergence.

The most pronounced difference between traditional and federated Boolean matrix factorization lies in data accessibility. Rather than having all data $A \in \{0,1\}^{n \times m}$ accessible at one location, the data A is given as (horizontally) partitioned matrices $A_1, \ldots, A_c$ over $C \in \mathbb{N}$ clients such that

$$A = \left[A_1, \cdots, A_C\right]^\top \ ,$$

where $A_j \in \{0,1\}^{n_j \times m}$ and $n = \sum_i n_i$. We aim to discover a single *shared* matrix $\widehat{V} \in \{0,1\}^{k \times m}$ containing shared feature components that are beneficial for all clients. Due to privacy restrictions, we are however neither permitted to transmit matrices A_i 'offsite' (including to any other device), nor are we allowed to be able to draw conclusions about where components belong to. We want to factorize the data $A_i \approx U_i \circ \widehat{V}$ in terms of *local* matrix $U_i \in \{0,1\}^{\frac{n}{C} \times k}$ (associating data to components), and one shared *global* matrix $\widehat{V} \in \{0,1\}^{k\times m}$ (associating features into components). Without the knowledge of U_i, we *cannot* estimate specific attributes of individual users (assuming sufficiently large client datasets). We *can*, however, estimate sets of commonly co-occurring attributes across all clients, e.g. common combinations of genetic markers that are indicative of a disease.

Locally computing U_i for given A_i and $\widehat{V}$ is a regular Boolean matrix factorization. However, computing the *shared* $\widehat{V}$ without access to A_i and U_i is not straightforward. To enable the computing of a shared factor while still preserving privacy, we split the problem into subproblems Φ_i, introducing a local *but shareable* coefficient matrix $V_i \in \{0,1\}^{k\times m}$. In a nutshell, we estimate factorizations

$$U_i, V_i \leftarrow \underset{U_i,V_i}{\arg\min}\, \Phi_i(U_i, V_i, \widehat{V})$$

for all client subproblems Φ_i, combine local matrices $V_i \in \{0,1\}^{k\times m}$ into a shared matrix $\widehat{V}$, update V_i, and repeat until convergence.

Local Subproblems and Clients

A single subproblem at client $i \in \mathbb{N}$, seeks to optimize $A_i \approx [U_i \circ V_i]_{ab} = \bigvee_{c\in[k]} U_{i,cl} V_{i,cb}$, of two low-rank Boolean factor matrices (Miettinen et al. 2008), $U_i \in \{0,1\}^{n_i\times k}$ (*feature matrix*) and $V_i \in \{0,1\}^{k\times m}$ (*coefficient matrix*). As this problem is NP-complete (Miettinen et al. 2008), solving it exactly is challenging for each client, even for relatively small matrices.

A major factor contributing to this hardness is the requirement that variables are Boolean. To address this challenge, we essentially relax the Boolean constraint by replacing it with additional penalty terms R and P detailed below. That is, we continuously relax the problem into a *binary matrix factorization* problem

$$\Phi_i(U_i, V_i) = \|A_i - U_i V_i\|_F^2 + R(U_i) + R(V_i) + P(V_i) \quad , \quad (1)$$

for relaxed $U_i \in [0,1]^{n_i\times k}$ and $V_i \in [0,1]^{k\times m}$ using regular linear algebra. First, To yield the desired Boolean outcomes without constrains, we introduce a *binary-inducing regularizer* $R : \mathbb{R}^{n'\times m'} \to \mathbb{R}$, enabling efficient gradient-based optimizations. A regularizer that encourages binary solutions combines two elastic-nets (rooted at 0 and 1, resp.) into the almost W-shaped ELB-regularizer

$$R_{\kappa\lambda}(X) = \min\{r(X), r(X - \mathbf{1})\} \quad (2)$$

where $r(x) = \kappa\|X\|_1 + {}^{\lambda}\!/\!_2\|X\|_2^2$ (Dalleiger and Vreeken 2022).

Second, for faster convergence towards a shared solution, we introduce a proximity penalty $P : \mathbb{R}^{n'\times m'} \to \mathbb{R}$, which encourages local V_i to remain close to the global model using the distance $P(V_i) = \gamma\|V_i - \widehat{V}\|_F^2$ between them.

Even though now unconstrained, this problem is still challenging due to being non-convex. We solve this joint objective by first splitting it in two subproblems, solving them using the alternating update

$$U_i^{t+1} = \underset{U}{\arg\min}\, \|A_i - UV_i^{t-1}\|_F^2 + R(U) \quad \text{and}$$

$$V_i^{t+1} = \underset{V}{\arg\min}\, \|A_i - U_i^{t+1}V\|_F^2 + R(V) + P(V) \ .$$

Because each individual objective remains a challenge due to the non-convexity, we require an optimization algorithm that is capable of solving such non-convex problems. To this end, we employ the *inertial proximal alternating linear minimization* (iPALM) technique (Pock and Sabach 2016), which will guarantee convergence (Attouch, Bolte, and Svaiter 2013; Bolte, Sabach, and Teboulle 2014) as detailed in Sec. 15.

Proximal Alternating Linear Minimization At the core of iPALM, each regularized objective for U_i and V_i are solved using a proximal gradient approach, which separates loss from regularizer. That is, after taking a gradient step concerning our linear least-squares loss f, e.g., $f(U) \leftarrow \|A_i - UV_i^{t-1}\|_F^2$, we then take a scaled proximal step regarding regularizer to project the gradient towards a feasible Boolean solution and towards a proximity to $\widehat{V}$ for V_i. A proximal operator is the projection

$$\text{prox}_\eta^R(X) = \underset{Y}{\arg\min}\, {}^1\!/\!_{2\eta}\|X - Y\|_F^2 + R(X) \quad (3)$$

of the result of the gradient step $x - x\eta\nabla_x f(x)$ for the loss f, into the proximity of a regularized solution $R(X)$. With regards to our regularizer R and P, these proximal problems lend themselves for deriving first-order optimal and efficiently-computable closed-form solutions: The *Boolean proximal operator* for R is element-wise computable

$$\text{prox}_\eta^R(X) = \frac{1}{1+\eta\lambda}\,\text{sign}(X-\theta)\max\{|X-\theta)| - \eta\kappa, 0\} \ , \quad (4)$$

for element-wise indicator $\theta = \mathbb{1}[X_{ij} \leq {}^1\!/\!_2]_{ij}$ (Dalleiger and Vreeken 2022), as shown in Apx. A. The $\widehat{V}$*-proximity proximal operator* for P is a weighted average, defined as

$$\text{prox}_\eta^P(X) = [1 + \eta\gamma]^{-1}(X + \eta\gamma\widehat{V}) \ , \quad (5)$$

Composing the two as $\text{prox}_\eta^{R,P} = \text{prox}_\eta^P \circ \text{prox}_\eta^R$ gives the alternating update rules

$$U_i^{t+1} = \text{prox}^R_{\eta^t_{U_i}}(U_i^t - \eta^t_{U_i}\nabla^t_{U_i}\|A_i - U_i^tV_i^t\|_F^2)$$

$$V_i^{t+1} = \text{prox}^{R,P}_{\eta^t_{V_i}}(V_i^t - \eta^t_{V_i}\nabla_{V_i}\|A_i - U_i^{t+1}V_i^t\|_F^2) \ . \quad (6)$$

To apply these rules, we require step sizes, utilizing linear nature of the loss, we propose two alternatives: first we use the gradient Lipschitz constant L for $\eta = {}^1\!/\!_L$, yielding the update rule for FELB. Second we employ Lee and Seung (2000)'s *multiplicative update rule* (MU) for NMF with step size matrices $\eta^t_{U_i} = U_i \oslash U_iV_iV_i^\top$ and $\eta^t_{V_i} = V_i \oslash U_i^\top U_iV_i$ using the Hadamard division $\oslash$, containing individual step sizes for all elements in U_i and V_i, yielding FELB$^{\text{MU}}$.

Global Objective and Server

Now having established our per client subproblems, we now combine the local subobjectives into one global objective Φ

$$\Phi(U, V, \widehat{V}) = \sum_i \Phi_i(U_i, V_i) + R(\widehat{V}) \ , \quad (7)$$

focusing on shared coefficients components $\widehat{V}$. To estimate the shared matrix $\widehat{V}$ *independent* of all data matrices A_i and local basis matrices U_i, we have to combine V_i matrices. In federated learning, this is often done by aggregating all V_i as the average $\widehat{V}$. However, doing so here does not necessarily yield valid results: naïve averaging results in aggregates that are far from being binary, thus hindering or even preventing convergence. Addressing this aggregation problem, we aim to result in a Boolean matrix, for which we iteratively project the aggregate towards a valid Boolean values

$$\widehat{V} \leftarrow \arg\min_{\widehat{V}} \sum_i {}^1\!/\!_2 \|\widehat{V} - V_i\|_{\mathrm{F}}^2 + R(\widehat{V}) \ , \tag{8}$$

for which we employ a proximal aggregation yielding the update-step $\widehat{V} \leftarrow \mathrm{prox}_{\eta}^{R} \frac{1}{c} \sum V_i$. To theoretically guarantee *differential privacy*, clients may further distort the matrices V_i before transmission, as described next.

Guaranteeing Differential Privacy

The proposed aggregation approach only shares coefficient matrices, so that no direct relationships between observations are shared. An attacker or a curious server can, however, attempt to infer private data from coefficients V_i. Aiming to prevent this, we guarantee differential privacy using an additive noise mechanisms, where, in a nutshell, each client adds noise before it transmits V_i to the server. We consider the Bernoulli, Gaussian, and Laplacian mechanisms, which only differ in the noise distribution. Using a Gaussian mechanism, we achieve (ϵ, δ)-differential privacy as follows.

Definition 3.1 (Dwork, Roth et al. (2014))**.** For $\epsilon, \delta > 0$, a randomized algorithm $\mathcal{A} : \mathcal{X} \rightarrow \mathcal{Y}$ is (ϵ, δ)-differentially private (DP) if

$$P(\mathcal{A}(X) \in S) \leq e^{\epsilon} P(\mathcal{A}(X') \in S) + \delta$$

holds for each subset $S \subset \mathcal{Y}$ and for all pairs of neighboring inputs X, X' .

Applying Gaussian noise with 0 mean and σ variance to the local coefficients V_i before sending ensures (ϵ, δ)-DP (Balle and Wang 2018) for $\sigma = \Delta\epsilon^{-1}\sqrt{2\log(5/(4\delta))}$, where $\Delta = \sup_{X,X'} \|\mathcal{A}(X) - \mathcal{A}(X')\|$ is the sensitivity of $\mathcal{A}$. To ensure bounded sensitivity, we clip all V_i with clipping threshold $\theta > 1$ (Noble, Bellet, and Dieuleveut 2022). Similarly, adding 0-mean $\Delta\epsilon^{-1}$-variance Laplacian noise achieves $(\epsilon, 0)$-DP (Dwork et al. 2006).

Convergence Analysis

Having ensured differential privacy, we summarize our algorithm. We call the combination of this proximal aggregation with local proximal-gradient optimization steps the FELB algorithm, detailed in Alg. 1: Local factors U_i, V_i are initialized uniformly at random (line 1), and at each client in round t (line 2), we update the local factor matrices (lines 4 and 5). Every b rounds, we transmit the local matrices V_i to the server (line 7). At this point, each client may choose to preserve differential privacy. The server, receives all local coefficients V_i (line 11), averages the matrices, and applies the proximal-operator (line 12). The aggregate is then transmitted to all clients (line 13). Upon receiving the aggregate (lines 8 and 9), each client continues with the next optimization round.

Algorithm 1: Federated Binary Matrix Factorization with FELB

Input: distributed target matrices $A^1, \ldots, A^C$, component count k
Output: local matrices $U_1, \ldots, U_C$, global coefficient matrix $\widehat{V}$

1 initialize U_i, V_i for $i \in [C]$ uniformly at random
2 **At client** *i at iteration t* **do**
3     $U_i \leftarrow \mathrm{prox}_{\eta_{U_i}}^{R_t}(U_i - \eta_{U_i}\nabla_{U_i}\|A_i - U_iV_i\|_{\mathrm{F}}^2)$
4     $V_i \leftarrow \mathrm{prox}_{\eta_{V_i}}^{R_t,P}(V_i - \eta_{V_i}\nabla_{V_i}\|A_i - U_iV_i\|_{\mathrm{F}}^2)$
5   **if** $t \bmod b = 0$ **then**
6   **if** *is differentially private* **then**
7             $V_i \leftarrow V_i \oplus N, N_{ab} \sim \mathcal{N}(0, \sigma)$
8   transmit V_i to the server
9   receive $\widehat{V}$ from the server
10   let $V_i \leftarrow \widehat{V}$
11 **At server do**
12   receive $V_1, \ldots, V_C$
13   aggregate $\widehat{V} \leftarrow \mathrm{prox}_{\eta}^{R_t}\left(\frac{1}{C}\sum_{i=1}^{C} V_i\right)$
14   transmit $\widehat{V}$ to each client
15 **return** U, $\widehat{V}$

Next, to formally ascertain that Alg. 1 solves our problem, we show that the algorithm converges with Thm. 3.2, and achieves Boolean coefficients in the limit with Thm. 3.3.

Theorem 3.2 (Convergence)**.** *For the sequence generated by Alg. 1* $\{z^t \triangleq (\{U_i^t\}_i, \{V_i^t\}_i, \bar{V}^t)\}_{t\in\mathbb{N}}$*, the objective function* $\Phi(z^t)$ *converges to a stable solution* $\Phi(z^t) \rightarrow \widehat{\Phi}$ *if* $t \rightarrow \infty$*.*

Proof. (Sketch, full proof in Apx. B). We show the objective's convergence to a stable solution Φ^* by initially establishing the convergence of each client, where we observe a *sufficient reduction* in local objectives, as well as a *bounded dissimilarity* to $\widehat{V}$. Leveraging this, we establish global convergence by showing that the global loss gradient is bounded by a *diminishing term*, showing that $\Phi(z^t)$ approaching a constant $\widehat{\Phi}$ as t tends to infinity. □

Theorem 3.3 (Boolean Convergence)**.** *If* λ^t *is a monotonically increasing sequence with* $\lambda^{t-1} \leq \lambda^t$, $\lim \lambda^t \rightarrow \infty$*, and* $\lambda^t - \lambda^{t-1} \leq \infty$*, then* $V_1^T, \ldots, V_c^T$ *and* $\widehat{V}^T$ *from the sequence generated by Alg. 1 convergences as* $\lim_{T\rightarrow\infty} \mathrm{dist}(\widehat{V}^T, \{0, 1\}) \rightarrow 0$ *to a Boolean matrix.*

Proof. (Sketch, full proof in Apx. B). Since gradients are bounded and diminish, we only need to show that the proximal operator returns Boolean solutions in the limit. As our gradients are Lipschitz continuous, bounded, and ensured to converge to a stable solution, our scaled proximal operator projects values onto Boolean results, for a monotonically increasing regularizer rate λ^t that approaches infinity in the limit, guaranteeing a stable Boolean convergence regardless of communication rounds. □

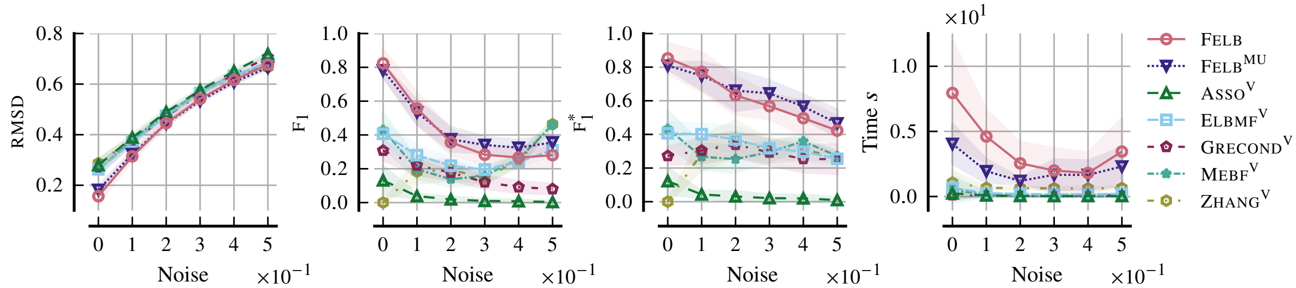


Figure 2: FELB and FELB$^{\mathrm{MU}}$ are robust against noise. We show the loss, recall, similarity, and elapsed runtime (s/C) for synthetic data with varying levels of destructive XOR noise.

Turning Traditional BMF into FedBMF

Given that there exist no federated matrix factorization algorithms tailored to binary data, we compare our approaches to local BMF methods, whose outcomes are then partially transmitted to a central location and collectively aggregated, following established ad-hoc federation strategies (Kamp 2019). In particular, we adapt the localized algorithms, covering the state of the art in the method families (1) *cover-based Boolean matrix factorizations* (ASSO, Miettinen et al. (2008); GRECOND, Belohlávek and Vychodil (2010); MEBF, Wan et al. (2020)) and (2) *relaxation-based binary matrix factorizations* (ZHANG, Zhang et al. (2007); and ELBMF, Dalleiger and Vreeken (2022)), to factorize *distributed* matrices—factorizing locally and aggregating the coefficient matrices centrally, replacing the local coefficients. Leveraging the following aggregations, we summarize the BMF federation scheme in Apx. C Alg. 2. To ensure binary results, we employ three *aggregation strategies* that maintain valid matrices

$$\text{Rounded Average} \quad \left\lfloor C^{-1} \textstyle\sum_{c\in[C]} V^c \right\rceil \tag{9}$$

$$\text{Majority Vote} \quad \left[\textstyle\sum_{c\in[C]} V^c_{ij} \geq C/2 \right]_{ij} \tag{10}$$

$$\text{Logical } \texttt{Or} \quad \bigvee V^1 \cdots V^C \tag{11}$$

We now describe our diverse set of experimental setups. First, we ascertain that FELB works reliably on synthetic data. Second, we empirically assess the differential-privacy properties of FELB. And third, we verify that FELB performs well on diverse real-world datasets drawn from four different scientific areas. To quantify the results, we report the *root mean squared deviation* (RMSD) and the F_1 *score* between data and reconstruction, as well as the runtime in seconds.

4 Experiments

We implement FELB in the Julia language and run experiments on 32 CPU Cores of an AMD EPYC 7702 or one NVIDIA A40 GPU, reporting wall-clock time in seconds. We provide the source code, datasets, synthetic dataset generator,[1] and additional information regarding reproducibility in Apx. E. In all experiments, we limit each algorithm run to 12h in total. We quantify the performance of *federated* ASSO, GRECOND, ELBMF, MEBF, FELB, and FELB$^{\mathrm{MU}}$ in terms of loss, recall, similarity, and runtime, reporting results for *majority voting* in the following, as it has superior performance to *rounded averaging* and *logical*, as shown in Apx. F.

[1] https://doi.org/10.5281/zenodo.14501661

Experiments on Synthetic Data

In these experiments, we answer the following questions:
Q1 How robust are the algorithms wrt. noise?
Q2 How scalable are they with increasing client counts?
Q3 How well do they perform under differential privacy?
To answer these, we need a controlled test environment. We construct this by sampling random binomial-noise matrices, into which we insert randomly generated, densely populated 'tiles' containing approximately 90% with 1s. To highlight trends, rather than random fluctuations, we report the mean and confidence intervals of 10 randomized trials.

Robustness regarding Noise To study the impact of noise on the quality of reconstructions, we generated synthetic matrices with varying degrees of destructive XOR noise, ranging from 0% (no noise, consisting solely of high-density tiles) to a maximum of 50% (completely random noise). Employing a fixed number of 10 clients, we applied federated ASSO, GRECOND, MEBF, ELBMF, and ZHANG, alongside FELB and FELB$^{\mathrm{MU}}$ to each dataset.

We present RMSD, F_1 score (re signal and noise data), F_1^* score (re signal), and runtime in Fig. 2: We see that reconstruction quality declines with increasing noise, yet FELB and FELB$^{\mathrm{MU}}$ achieve the best reconstructions across the board even at high noise levels. We see that RMSD and F_1 follow a similar trend across all methods, yet our methods consistently outperform the rest. However, if we regard only the interesting data signal with F_1^*, we see that FELB and FELB$^{\mathrm{MU}}$ are the only algorithms that result in good reconstructions of the ground-truth signal, even under high noise. This shows the ability of FELB and FELB$^{\mathrm{MU}}$ to discern pure noise from meaningful signal. While the runtime of ASSO, GRECOND, MEBF, ZHANG, and ELBMF is slightly faster in Fig. 2 (right), FELB$^{\mathrm{MU}}$'s and FELB's runtime reduces with increasing noise.

Scalability regarding Clients Next, we analyze the scalability of federated ASSO, GRECOND, ELBMF, MEBF, and

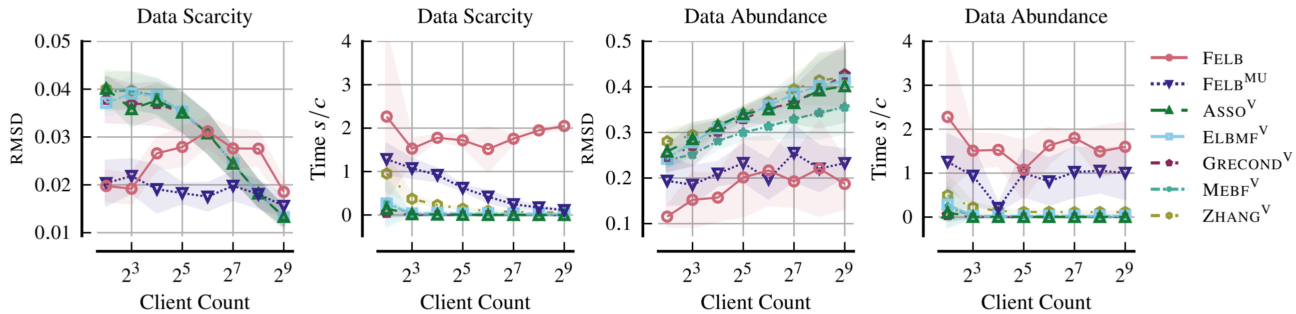


Figure 3: Felb and Felb$^{\text{MU}}$ perform well across various client counts, showing RMSD and runtime (s/C). For *data scarcity*, we fix the data size and an increase number of clients. For *data abundance* we grow data while increasing the number of clients.

Zhang under *majority voting*, as well as of Felb and Felb$^{\text{MU}}$, for varying numbers of clients, considering two contrasting scenarios of *scarce* and *abundant* data. In both cases, we generate and uniformly distribute synthetic data to a number of clients, depicting results in Fig. 3.

To create *data scarcity*, we fix the dataset size to 2^{16} and increase the number of clients from 2^2 to 2^9, thus iteratively reducing the sample count per client. In Fig. 3 (left), we observe that our methods scale well to low-sample scenarios and deliver the best performance. The MU update rule outperforms the competitors. The runtime of post-hoc federated methods Asso, Grecond, mebf, Zhang, and Elbmf is lower since they only perform a single optimization epoch. These methods slightly outperform Felb and Felb$^{\text{MU}}$ only in tiny data scenarios where the estimator-variance is high, while the Felb$^{\text{MU}}$ significantly outperforms all methods and is notably faster than Felb.

To evaluate under *data abundance*, we scale the number of samples by increasing the number of clients from 2^2 to 2^9, maintaining a constant sample count of 500 per client. In Fig. 3 (right), we observe that our methods scale well with an increased number of clients. With more data, Felb using Lipschitz steps slightly outperforms the MU steps in RMSD, and both methods exhibit comparable runtime trends. The runtime of post-hoc federated methods Asso, Grecond, mebf, Zhang, and Elbmf remains lower, as they compute only one local optimization epoch.

Performance under Privacy To empirically ascertain the effect of differential-privacy on the loss, we add noise to the transmitted factor matrices according to various noise mechanisms. Specifically, we study the effect of additive clipped or regular Laplacian and Gaussian, as well as xor Bernoulli noise mechanisms, as depicted in Fig. 4 and Apx. F, for varying $0 \leq \epsilon \leq 2$ and fixed $\delta = 0.05$. Because Asso, mebf, Grecond, Zhang, and Elbmf return Booelean matrices, we subject these only to xor noise, rather than additive noise, to retain Boolean matrices. The results in Fig. 4 show that both Felb and Felb$^{\text{MU}}$ exhibit similar performance across various noise models, while Felb$^{\text{MU}}$ is most robust. The plots display three phases: In the low-ϵ domain, there is almost no performance deterioration, followed by a steep, hockey-stick-like descent which eventually stabilizes in the high-ϵ range. We note an increasing 'sharpness' of the hockey-stick-phase under clipping, showing less smooth reactions to privacy adjustments for both mechanisms.

Experiments on Real-World Data

Having established the efficiency and precision of our method on synthetic data, we proceed to assess its effectiveness on real-world datasets. For this, we curated a diverse set of 8 real-world datasets spanning four distinct domains. To explore **recommendation systems**, we include *Goodreads* (Kotkov et al. 2022) for books and *Movielens* (Harper and Konstan 2015) and *Netflix* (Netflix, Inc. 2009) for movies, where user ratings $\geq$ 3.5 are binarized to 1. In **life sciences**, we use *TCGA* (Institute 2005) for cancer genomics, *HPA* (Bakken et al. 2021; Sjöstedt, Zhong, and et. al 2020) for single-cell proteomics, and *Genomics* (Oleksyk, Gonçalo, and et. al 2015) for mutation data. *TCGA* marks gene expressions in the top 95% quantile as 1, while *HPA* designates observed RNA in cells as 1. For **social science**, we analyze poverty *(Pov)* and income *(Inc)* using the *ACS* (U.S. Census Bureau 2023) dataset, binarizing with one-hot encoding utilizing Folktables (Ding et al. 2021). In **natural language processing**, we study higher-order word co-occurrences in ArXiv cs.LG abstracts (Collaboration 2023). Each paper abstract is a row with columns marked 1 if the corresponding word is in the vocabulary, containing lemmatized, stop-word-free words with a minimum frequency of 1 ‱ . We summarize dataset extents, density, and chosen component counts in Apx. E, Tbl. 2. Since the number of clients (e.g., Hospitals) is expected to be small, we limit the federation to a reasonable $C = 50$ clients, on which we compare federated methods Asso, Grecond, mebf, Elbmf, and Zhang, as well as Felb, and Felb$^{\text{MU}}$ across all real-world datasets, synchronizing after every $b = 10$ local optimization rounds.

In Tbl. 1, we present the F_1 between the reconstruction and the data matrix, where – indicate missing data due to time limits. Our results show that Felb and Felb$^{\text{MU}}$ exhibit best-in-class performance, consistently ranking as the best or second-best algorithms. This performance gap is evident across all datasets except for the *HPA* dataset, where mebf, a method designed with similar data types in mind, outperforms the others, and the *ACS Pov* dataset, where Grecond

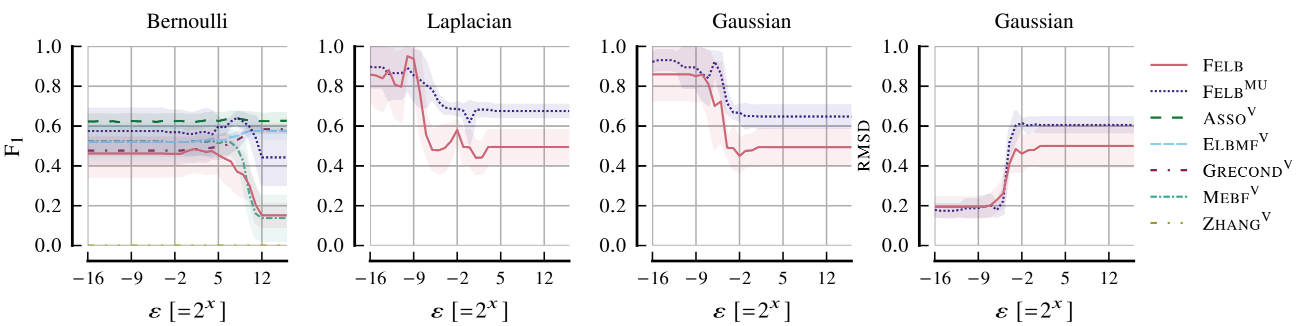


Figure 4: FELB and FELB^{MU} achieve accurate yet differentially private reconstructions. For synthetic data, we subject algorithms to different noise mechanisms: Bernoulli, Laplacian, and Gaussian noise.

Dataset	ASSO^V	MEBF^V	GRECOND^V	ZHANG^V	ELBMF^V	FELB^{MU}	FELB
ACS Inc	0.388	0.108	**0.690**	0.000	0.000	0.585	0.328
ACS Pov	0.692	–	**0.797**	0.000	0.217	0.638	0.517
cs.LG	–	0.000	**0.068**	0.000	0.000	0.057	0.006
Goodreads	–	0.000	0.017	–	0.000	**0.125**	0.059
HPA Brain	–	**0.642**	–	0.000	0.000	0.007	0.000
Movielens	–	0.017	–	–	0.000	**0.193**	0.163
Netflix	–	0.010	–	–	0.000	**0.197**	0.144
TCGA	0.039	0.055	0.007	0.000	0.000	**0.414**	0.402
Avg. Rank	4.750	3.75	3.375	5.125	4.500	**1.625**	2.750

Table 1: FELB and FELB^{MU} consistently perform well. We illustrate the F_1 of ASSO, GRECOND, MEBF, ELBMF, and ZHANG under voting aggregation, as well as federated FELB, and FELB^{MU} on 8 real-world data across 50 clients. We highlight the best scoring algorithm with **bold**, the second best with underline, and timeouts by a dash –.

leads. Notably, since clients of ELBMF and ZHANG diverge significantly, they often aggregate into a no-consensus 0-only global model matrix, thus showing low accuracy. Although they perform only a single optimization round per client, we see that ASSO, GRECOND, and MEBF do not finish on medium to large datasets. Additionally, we show the RMSD in Apx. F, where FELB and FELB^{MU} are on top, and compare client-server communication frequencies in Apx. F, demonstrating the strength of FELB and resp. FELB^{MU}.

5 Discussion and Conclusion

We introduced the federated proximal-gradient-based FELB for BMF tasks, showed its convergence to a binary outcome in theory, and demonstrated its efficacy in experimental practice. We provided a variant called FELB^{MU}, whose practical performance outcompetes FELB on many real-world datasets, especially under rare synchronizations. Although FELB and FELB^{MU} perform consistently well, both are first-of-their-kind federated BMF algorithms. As such, they leave ample room for further research.

Limitations Our research focuses on learning from private Boolean data generated by similar sources at a few research centers, thus we concentrate on suitable experiments and abstain from distant but related problems, such as learning with millions of heterogeneous clients. Further, we experimentally demonstrate the practical limitations of our methods.

Future Work includes extending FELB to allow for heterogeneous clients and data distributions, adapting our methods to learn from varied data distributions and characteristics. Additionally, we plan to explore large-scale federations, drawing inspiration from frameworks like Scaffold (Karimireddy et al. 2020) and FedProx (Li et al. 2020b) for efficient client sampling, variance controlling, and formal limits to client dropout resilience. Furthermore, we intend to investigate personalized federated learning techniques to improve the reconstructions in case of varied data sources. Finally, we plan to move beyond Boolean data and seek explore the potential of allowing partial sharing of a subset of the client components V_i to allow for multi-source multi-modal federated learning to improve model performance and generality.

Acknowledgements

This research is supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Michael Kamp received support from the Cancer Research Center Cologne Essen (CCCE).

References

Araujo, M.; Ribeiro, P. M. P.; and Faloutsos, C. 2016. FastStep: Scalable Boolean Matrix Decomposition. In *Advances in Knowledge Discovery and Data Mining - 20th Pacific-Asia Conference, PAKDD 2016, Auckland, New Zealand, April 19-22, 2016, Proceedings, Part I*, volume 9651 of *Lecture Notes in Computer Science*, 461–473. Springer.

Attouch, H.; Bolte, J.; and Svaiter, B. F. 2013. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Math. Program.*, 137(1-2): 91–129.

Bakken, T. E.; Jorstad, N. L.; Hu, Q.; and et. al, B. B. L. 2021. Comparative cellular analysis of motor cortex in human, marmoset and mouse. *Nature*, 598: 111 – 119.

Balle, B.; and Wang, Y.-X. 2018. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, 394–403. PMLR.

Belohlávek, R.; and Vychodil, V. 2010. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *Journal of Computer and System Sciences*, 76(1): 3–20.

Bolte, J.; Sabach, S.; and Teboulle, M. 2014. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2): 459–494.

Collaboration, A. 2023. arXiv dataset and metadata of 1.7M+ scholarly papers across STEM.

Dalleiger, S.; and Vreeken, J. 2022. Efficiently Factorizing Boolean Matrices using Proximal Gradient Descent. *Advances in Neural Information Processing Systems*.

Ding, F.; Hardt, M.; Miller, J.; and Schmidt, L. 2021. Retiring Adult: New Datasets for Fair Machine Learning. *Advances in Neural Information Processing Systems*, 34.

Du, Y.; Zhou, D.; Xie, Y.; Shi, J.; and Gong, M. 2021. Federated matrix factorization for privacy-preserving recommender systems. *Applied Soft Computing*, 111: 107700.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, 265–284. Springer.

Dwork, C.; Roth, A.; et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4): 211–407.

Golub, G. H.; and Loan, C. F. V. 1996. *Matrix Computations, Third Edition*. Johns Hopkins University Press. ISBN 978-0-8018-5414-9.

Haddad, A.; Shamsi, F.; Zhu, L.; and Najafizadeh, L. 2018. Identifying Dynamics of Brain Function Via Boolean Matrix Factorization. In *52nd Asilomar Conference on Signals, Systems, and Computers, ACSSC 2018, Pacific Grove, CA, USA, October 28-31, 2018*, 661–665. IEEE.

Harper, F. M.; and Konstan, J. A. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.*, 5(4).

Hess, S.; and Morik, K. 2017. C-SALT: Mining Class-Specific ALTerations in Boolean Matrix Factorization. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22 , 2017, Proceedings, Part I*, volume 10534 of *Lecture Notes in Computer Science*, 547–563. Springer.

Hess, S.; Morik, K.; and Piatkowski, N. 2017. The PRIMPING routine - Tiling through proximal alternating linearized minimization. *Data Mining and Knowledge Discovery*, 31(4): 1090–1131.

Ignatov, D. I.; Nenova, E.; Konstantinova, N.; and Konstantinov, A. V. 2014. Boolean Matrix Factorisation for Collaborative Filtering: An FCA-Based Approach. In *Artificial Intelligence: Methodology, Systems, and Applications - 16th International Conference, AIMSA 2014, Varna, Bulgaria, September 11-13, 2014. Proceedings*, volume 8722 of *Lecture Notes in Computer Science*, 47–58. Springer.

Institute, N. C. 2005. The Cancer Genome Atlas Program (TCGA).

Kamp, M. 2019. *Black-box parallelization for machine learning*. Ph.D. thesis, Universitäts-und Landesbibliothek Bonn.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; and Suresh, A. T. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, 5132–5143. PMLR.

Khanna, R.; Zhang, L.; Agarwal, D.; and Chen, B.-C. 2013. Parallel matrix factorization for binary response. In *2013 IEEE International Conference on Big Data*, 430–438. IEEE.

Kotkov, D.; Medlar, A.; Maslov, A.; Satyal, U. R.; Neovius, M.; and Glowacka, D. 2022. The Tag Genome Dataset for Books. In *Proceedings of the 2022 ACM SIGIR Conference on Human Information Interaction and Retrieval (CHIIR '22)*. ISBN 978-1-4503-9186-3/22/03.

Lee, D. D.; and Seung, H. S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755): 788–791.

Lee, D. D.; and Seung, H. S. 2000. Algorithms for Non-negative Matrix Factorization. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, 556–562. MIT Press.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020a. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2: 429–450.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020b. Federated Optimization in Heterogeneous Networks. In Dhillon, I. S.; Papailiopoulos, D. S.; and Sze, V., eds., *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*. mlsys.org.

Li, Z.; Ding, B.; Zhang, C.; Li, N.; and Zhou, J. 2021. Federated matrix factorization with privacy guarantee. *Proceedings of the VLDB Endowment*, 15(4).

Liang, L.; Zhu, K.; and Lu, S. 2020. BEM: Mining Coregulation Patterns in Transcriptomics via Boolean Matrix Factorization. *Bioinformatics*, 36(13): 4030–4037.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, 1273–1282. PMLR.

Miettinen, P.; Mielikäinen, T.; Gionis, A.; Das, G.; and Mannila, H. 2008. The Discrete Basis Problem. *IEEE Transactions on Knowledge and Data Engineering*, 20(10): 1348–1362.

Mishchenko, K.; Malinovsky, G.; Stich, S.; and Richtarik, P. 2022. ProxSkip: Yes! Local Gradient Steps Provably Lead to Communication Acceleration! Finally! In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 15750–15769. PMLR.

Nesterov, Y. 2004. *Introductory Lectures on Convex Optimization*. Springer US. ISBN 9781441988539.

Netflix, Inc. 2009. Netflix Prize.

Neumann, S.; and Miettinen, P. 2020. Biclustering and Boolean Matrix Factorization in Data Streams. *Proceedings of the VLDB Endowment*, 13(10): 1709–1722.

Noble, M.; Bellet, A.; and Dieuleveut, A. 2022. Differentially private federated learning on heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, 10110–10145. PMLR.

Oleksyk, T. K.; Gonçalo, A.; and et. al, R. D. 2015. A global reference for human genetic variation. *Nature*, 526: 68–74.

Paatero, P.; and Tapper, U. 1994. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2): 111–126.

Parikh, N.; and Boyd, S. P. 2014. Proximal Algorithms. *Foundations and Trends in Optimization*, 1(3): 127–239.

Pock, T.; and Sabach, S. 2016. Inertial Proximal Alternating Linearized Minimization (iPALM) for Nonconvex and Nonsmooth Problems. *SIAM Journal on Imaging Sciences*, 9(4): 1756–1787.

Sjöstedt, E.; Zhong, W.; and et. al, L. F. 2020. An atlas of the protein-coding genes in the human, pig, and mouse brain. *Science*, 367(6482): eaay5947.

U.S. Census Bureau. 2023. Census. U.S. Department of Commerce.

Wan, C.; Chang, W.; Zhao, T.; Li, M.; Cao, S.; and Zhang, C. 2020. Fast and Efficient Boolean Matrix Factorization by Geometric Segmentation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, 6086–6093. AAAI Press.

Yu, H.-F.; Hsieh, C.-J.; Si, S.; and Dhillon, I. S. 2014. Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, 41: 793–819.

Zhang, Z.; Li, T.; Ding, C. H. Q.; and Zhang, X. 2007. Binary Matrix Factorization with Applications. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA*, 391–400. IEEE Computer Society.