CometCloudCare (C^3): Distributed Machine Learning Platform-as-a-Service with Privacy Preservation

Vamsi K. Potluru
Rutgers Discovery Informatics Institute
Rutgers University
vamsi.potluru@rutgers.edu

Javier Diaz-Montes
Rutgers Discovery Informatics Institute
Rutgers University
javidiaz@rdi2.rutgers.edu

Anand D. Sarwate
Dept. of Electrical and Computer Engineering
Rutgers University
asarwate@ece.rutgers.edu

Sergey M. Plis **Vince D. Calhoun**
Mind Research Network
splis@mrn.org vcalhoun@mrn.org

Barak A. Pearlmutter
Hamilton Institute
National University of Ireland Maynooth
barak@cs.nuim.ie

Manish Parashar
Dept. of Computer Science
Rutgers University
parashar@rutgers.edu

1 Introduction

The growth of data sharing initiatives in neuroscience and genomics [14, 16, 19, 25] represents an exciting opportunity to confront the "small N" problem plaguing contemporary studies [20]. When possible, open data sharing provides the greatest benefit. However some data cannot be shared at all due to privacy concerns and/or risk of re-identification. Sharing other data sets is hampered by the proliferation of complex data use agreements (DUAs) which preclude truly automated data mining. These DUAs arise because of concerns about the privacy and confidentiality for subjects; though many do permit direct access to data, they often require a cumbersome approval process that can take months. Additionally, some researchers have expressed doubts about the efficiency and scalability of centralized data storage and analysis for large volume datasets [18]. In response, distributed cloud solutions have been suggested [23]; however, the task of transferring large volumes of imaging data (processed or unprocessed) to and from the cloud is far from trivial. More worrisome than the challenges of data transfer and storage is the tendency for labs to collect, label, and maintain neuroimaging data in idiosyncratic ways. Developing standardized data collection and storage is a recent trend [26], and achieving such a standard may take years, or may never happen at all.

Consider a psychiatric researcher who wishes to understand the neurophysiological differences between patients with schizophrenia and those without. Specifically, this researcher wishes to discover latent features in structural magnetic resonance imaging (sMRI) brain scans, which are relevant for distinguishing the healthy control subjects from those with the disorder. In order to robustly discover these latent features, the researcher would need access to a much larger population dataset than that typically available at a single site. Rather than developing an entirely new study with new subjects at great cost, it would be desirable to re-use data produced by other researchers who are studying or have studied schizophrenia. This will not only reduce the cost of a particular study, but will also gain access to a data set of sufficient size. The researcher may be familiar with current statistical and machine learning methods, such as support-vector machine classification, independent components analysis (ICA), or nonnegative matrix factorization (NMF), but likely has limited experience with distributed data mining, cloud computing, or using computational infrastructures more complex than a desktop personal computer.

The system we propose is a distributed service which seamlessly provides transparent access to resources, matching the requirements of the researcher while preserving data privacy. The researcher would merely specify the datasets to use, a criteria to select specific subjects from those datasets, and possibly some data privacy constraints, such as data movement or a privacy budget.[1] They would also select a machine learning method from those available in the catalogue to be used in the data analysis—for example, latent feature discovery *via* nonnegative matrix factorization. The system then autonomously executes the selected machine learning method in a distributed fashion across the available infrastructure, subject to budgetary constraints, and returns the results to the researcher. By automating the computational resources, data governance, and other system implementation issues, the researcher is insulated from the details of how their request is processed. The $\mathbf{C^3}$ platform proposed here is design to provide a suite of functionalities which forms a kind of application programming interface (API) for researchers who may lack the technical expertise to leverage cloud computing and other technologies. Similarly, $\mathbf{C^3}$ can be extended to account for budgetary constraints such as privacy requirements for data holders, eliminating the need to traverse a Procrustean maze of individual per-use DUAs. Finally, this framework can take advantage of heterogeneous computational resources in an elastic and on-demand manner, leading to much more efficient processing. There is substantial value to performing data analysis on the fly, as the flow of data continues to increase. However, this does not mesh well with traditional approaches in which a dataset is under the complete control of the investigator and analyses take weeks, months, or longer to perform, re-check, and validate.

One of the motivations for $\mathbf{C^3}$ is the need to guarantee privacy in applications involving sensitive data. A canonical example of this is medical research: due to ethical and legal concerns, many data holders are hesitant to share raw data across the network, and would prefer a more gated access via trusted privacy-preserving algorithms. Here we take a step towards distributed private analyses by keeping the data at each site and sharing only summaries of local data sets. Such a system does not provide strong quantifiable privacy guarantees such as differential privacy. However, differential privacy is guaranteed by the algorithms, and our focus here is on the system design. We leave the problem of designing a differentially private version of our NMF algorithm for future work; differentially private algorithms for tasks such as classification have shown promising initial results, and we believe these advances can be incorporated easily into CometCloud. To our knowledge, no system for neuroimaging research is currently focused on advanced solutions to the competing goals of data analysis and privacy, beyond simple anonymization or "defacing" [2].

Our contributions. Our contribution in this paper is a framework and interface for researchers who work with sensitive data to run complex distributed machine learning algorithms on distributed data sets in a way that is transparent to the user. We demonstrate the feasibility of this task by implementing a distributed version of a nonnegative matrix factorization (NMF) algorithm [1, 11]. Such algorithms are attractive for nonnegative datasets since the latent factors can be directly interpreted by the domain expert. For instance, in structural MRI datasets, the input to the matrix factorization is gray matter concentration maps which are necessarily nonnegative. If both latent factors are nonnegative then the presence of a certain latent feature can be simply checked by looking at the corresponding element in the activation matrix. In contrast, independent component analysis (ICA) leads to real-valued features and activation patterns which are rarely interpretable .

The benefits of this approach are: (i) Transparent use of ML algorithms allows users to focus on specifying the algorithms they would like to run and the datasets to use. Datasets are selected based on criteria from their research study and are subjected to budgetary, system usage, and privacy constraints. The framework autonomously marshals appropriated resources and orchestrates the execution of the selected algorithms; (ii) Transparently develop ML algorithms, users focus on the logic of the machine learning algorithms without worrying about the underneath orchestration mechanisms; (iii) Privacy preservation control where users can add or use policies, such as differentially private algorithms with budget tracking.

2 CometCloud

CometCloud is an autonomic framework for enabling real-world applications on software-defined federated cyber-infrastructure, including heterogeneous infrastructures integrating public clouds (e.g., Amazon EC2), private clouds (e.g., private OpenStack deployment), and data-centers. CometCloud exposes the federated infrastructure using cloud abstractions to offer resources in an elastic and on-demand way, regardless their location. It also provides abstractions and mechanisms to support a range of programming paradigms and applications requirements on top of the federation [5]. The ultimate goal of such as system is to autonomously control the computation, storage, and communication

[1] Privacy budgets are an accounting mechanism for loss of privacy, monetary compensation, or other quantifiable constraints .

aspects of distributed data processing, leaving end users to merely specify what they want to do (e.g., process a dataset using a specific application) rather than how (i.e., how many computers and from which location, where to move data, how to interact with a specific system, etc.).

Conceptually, CometCloud is composed of a programming layer, service layer, and infrastructure layer. The infrastructure layer is composed by a dynamic self-organizing overlay which connects available resources. New resources can be added to the overlay at runtime or remove when they are not needed. This layer is fault tolerant and resilient to disconnects and failures. This layer also has a routing engine that allows to address resources using attributes (e.g., type of CPU, amount of memory) instead of specific addresses. Other features such as data replication, load balancing, notifications and event propagation are also supported. The service layer provides a range of services to support autonomics at the programming and application level. This layer provides management of application executions, discovery of services, associative object store to explicitly exploit context locality, and messaging service including publish/subscribe and push/pull. The programming layer provides the basic functionality for application development and management. It supports a range of paradigms including the master-worker and bag of tasks. Masters generate tasks and workers consume them. Scheduling and monitoring of tasks are transparently supported by the framework without users interaction. A task consistency service handles lost/failed tasks. CometCloud is not restricted to applications that have been developed in a particular programming language (e.g., Java), and has been demonstrated to work as a wrapper for a number of other languages (C, Matlab, Fortran, Python, and Scala) [4].

2.1 CometCloud Federation Model

The CometCloud federation model is based on the Comet [12] coordination spaces concept. A Comet spaces is, in essence, an overlay that is used to coordinate different aspects of the federation. In particular, we have decided to use two kind of spaces in the federation. First, we have a single federated management space used to create the actual federation and orchestrate the different resources. This space is used to interchange any operational message for discovering resources, announcing changes in a site, routing users request to the appropriate sites, or initiating negotiations to create ad-hoc execution spaces. On the other hand, we can have multiple shared execution spaces that are created on demand to satisfy computing needs of the users. Execution spaces can be created in the context of a single site to provision local resources and cloudburst to public clouds or external HPC systems. Moreover, they can be used to create a private sub-federation across several sites. This case can be useful when several sites have some common interest and they decide to jointly target certain type of tasks as a specialized community. The same mechanism can be used to set boundaries across sites controlling data movement or access to computation [4].

In the model, users at every site have access to a set of heterogeneous and dynamic resources, such as public/private clouds, supercomputers, and grids. These resources are uniformly exposed using cloud-like abstractions and mechanisms that facilitate the execution of applications across the resources. The federation is dynamically created in a collaborative way, where sites "talk" to each other to identify themselves, negotiate the terms of adhesion, discover available resources, and advertise their own resources and capabilities. Sites can join and leave at any point. Notably, this requires a minimal configuration at each site that amounts to specifying the available resources, a queuing system or a type of cloud, and credentials. As a part of the adhesion negotiation, sites may have to verify their identities using security mechanisms such as X.509 certificates or public/private key authentication.

3 Platform Design

In this paper we propose a platform, called CometCloudCare ($\mathbf{C^3}$) to enable the use and development of distributed machine learning algorithms that can take advantage of geographically distributed resources. In this platform privacy raises as a first class citizen ensuring that distributed machine learning algorithms make use of various data without compromising it. We envision users with two roles: regular users and power users. Regular users simply select a desired machine learning algorithm and privacy policy from the catalogue and specify what datasets they want to use. Users can always specify data movement constraints (e.g., a dataset cannot be moved outside of its location or it can only be moved within a region). The underlying system takes care of provisioning resources, moving data, and delivering the results. On the other hand, we envision power users that can create new machine learning algorithms by specifying different functions and a workflow that determines how data flows from one function to another. Similarly privacy policies can be created by specifying the functions and operations that need to be applied to data at any step of the workflow. This platform builds on top of CometCloud which enables access to elastically federated resources by presenting them as a single pool of resources. The architecture of our platform is presented in Figure 1.

At the top of our framework we have the platform layer, where users are presented with a catalogue of machine learning algorithms and privacy policies. In this way, users only need to provide basic information, such as: a workflow describing how different machine learning algorithms will be using the data; objectives and policies relative to the application (e.g., deadline, budget); and data sources. Then, the platform interacts with CometCloud to ensure that the computation is done following the requirements. Hence, user is kept away from low level details regarding where and how data is process. Additionally, the platform layer exposes appropriated APIs to allow power users creating new algorithms and policies.

The platform layer interacts with CometCloud to gain transparent access to heterogeneous resources (see Figure 1).As we described in Section 2, CometCloud is able to federate highly heterogeneous and dynamic cloud/grid/HPC infrastructures. These resources are presented as a single pool of resources using cloud-like capabilities. This enables the integration of public/private clouds and autonomic cloudbursts, i.e., dynamic scale-out to clouds to address extreme requirements such as heterogeneous and dynamics workloads, and spikes in demands.

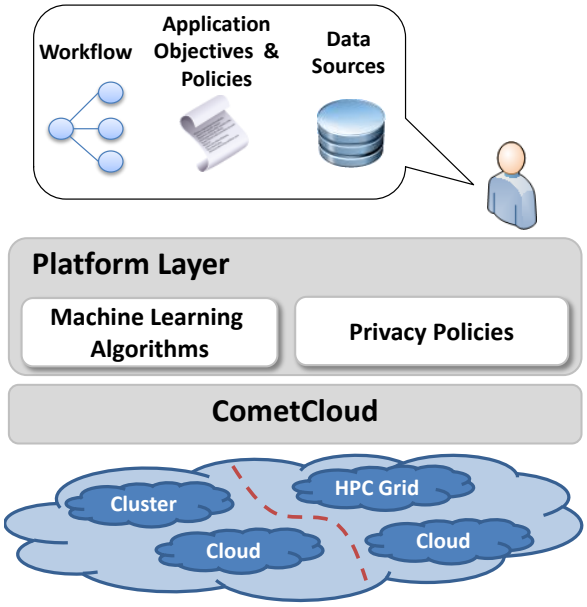

Figure 1: Platform-as-a-Service

3.1 The use case revisited

Let us consider a power-user, who wants to analyze structural MRI data spread across different medical centers. Assume that the machine learning model the researcher wants to apply is not in the list of implemented algorithms. The researcher can chose to implement a distributed version of the algorithm using the map/reduce paradigm. Our platform has APIs that allow users to create mapper and reducer functions which compute the parameters of the model when given well-defined data input and intermediary variables. Once the algorithm is integrated in the platform, users simply need to specify how input, outputs and intermediary results are used by these functions in a high level specification language, such as YAML. Using this information the system automatically handles task generation, scheduling, data movement, privacy requirements, etc. In particular, we have implemented in our platform the distributed nonnegative matrix factorization algorithm described in Section 4.2 .

4 Algorithm

Notation: Matrices will be denoted in boldface. The matrix $\mathbf{I}_d$ is the $d \times d$ identity matrix; the subscript will be omitted when the dimension is obvious. The notation $[n] = \{1, 2, \ldots, n\}$.

4.1 Nonnegative matrix factorization

Let the data be $\mathbf{X} \in \mathbb{R}^{d \times N}$ where d is the number of dimensions of each data point and N is the number of individuals. The data is assumed to be bounded and nonnegative, so that $\mathbf{X} \in [0,1]^{d \times N}$.

Orthogonal NMF seeks to perform a decomposition of the data matrix $\mathbf{X}$ into a product of two nonnegative matrices:

$$\mathbf{X} = \mathbf{AS} \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{d \times m}$ is a matrix whose columns are basis vectors, $\mathbf{S} \in \mathbb{R}^{m \times N}$ is a matrix of coefficients, and the two matrices satisfy the following:

$$\mathbf{A}^\top \mathbf{A} = \mathbf{I}_m \tag{2}$$

$$A_{ij} \geq 0 \qquad (i,j) \in [d] \times [k] \tag{3}$$

$$S_{ij} \geq 0 \qquad (i,j) \in [k] \times [N] \tag{4}$$

Because the data is bounded, the two matrices $\mathbf{A}$ and $\mathbf{S}$ are also bounded.

The orthogonal NMF algorithm is alternates the following two steps [3]:

$$\mathbf{S} \leftarrow \mathbf{S} \odot \frac{\mathbf{A}^\top \mathbf{X}}{\mathbf{A}^\top \mathbf{AS}} \tag{5}$$

$$\mathbf{A} \leftarrow \mathbf{A} \odot \frac{\mathbf{XS}^\top}{\mathbf{ASX}^\top \mathbf{A}} \tag{6}$$

where $\odot$ and matrix division are taken element-wise (Hadamard).

4.2 Distributed NMF

We assume the total data $\mathbf{X}$ is partitioned into K sites so that $\mathbf{X} = [\mathbf{X}_1 \ \mathbf{X}_2 \cdots \ \mathbf{X}_K]$. The distributed NMF algorithm can be decomposed into two stages corresponding to Map and Reduce phases of the MapReduce framework for large-scale learning. The Map phase involves learning the columns of matrix $\mathbf{S}$ corresponding to the current estimate of $\mathbf{A}$. Since the objective is parallelizable across the columns, each site can compute the corresponding columns of $\mathbf{S}$ given the corresponding data columns of $\mathbf{X}$.

Formally, the Map step for each site k we do the following update based on the local data $\mathbf{X}_k$ and the global variable $\mathbf{A}$:

$$\mathbf{S}_k \leftarrow \mathbf{S}_k \odot \frac{\mathbf{A}^\top \mathbf{X}_k}{\mathbf{A}^\top \mathbf{AS}_k} \tag{7}$$

Each site k produces two matrices, $\mathbf{S}_k$ and $\mathbf{X}_k \mathbf{S}_k^\top$. For the Reduce step, each site k transmits its estimate $\mathbf{X}_k \mathbf{S}_k^\top$ to a trusted central server which computes the sum

$$\mathbf{C} \leftarrow \sum_{k=1}^{K} \mathbf{X}_k \mathbf{S}_k^\top. \tag{8}$$

The Reduce step is given by:

$$\mathbf{A} \leftarrow \mathbf{A} \odot \frac{\mathbf{C}}{\mathbf{AC}^\top \mathbf{A}} \tag{9}$$

The matrix $\mathbf{A}$ is then sent to the individual sites. In this paper we develop a system for implementing this distributed NMF algorithm as well a differentially private version of the $\mathbf{A}$ update.

5 Experiments

We use a combined data from four separate schizophrenia studies conducted at Johns Hopkins University (JHU), the Maryland Psychiatric Research Center (MPRC), the Institute of Psychiatry, London, UK (IOP), and the Western Psychiatric Institute and Clinic at the University of Pittsburgh (WPIC) (the data used in [15]). The combined sample comprised 198 schizophrenia patients and 191 matched healthy controls and contained both first episode and chronic

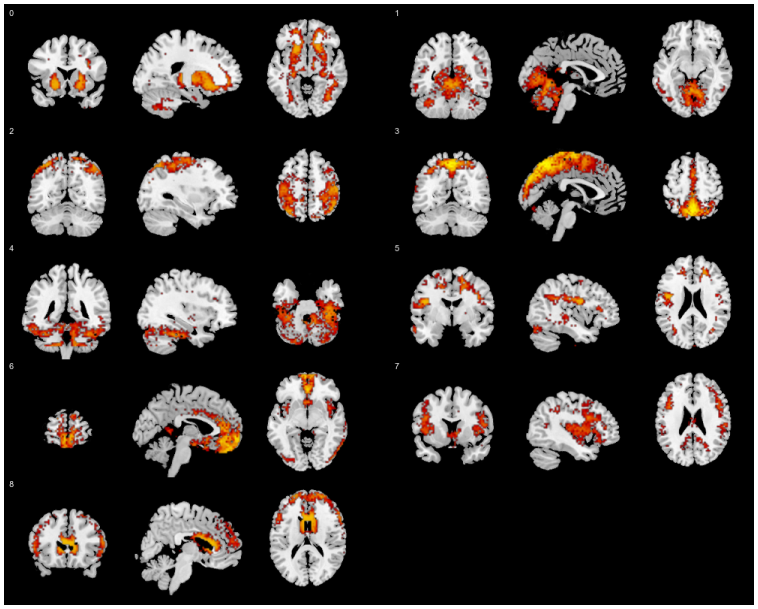

Figure 2: Shown are 9 features learnt using orthogonal NMF on a structural MRI dataset of 382 subjects with equal number of healthy controls and schizophrenic patients. The number of reduce steps is 10. We split the data evenly across two sites.

patients [15]. At all sites, whole brain MRIs were obtained on a 1.5T Signa GE scanner using identical parameters and software.

The distributed infrastructure consisted of two sites: (i) Rutgers Federation site: The Rutgers federation site is deployed on a cluster-based infrastructure with 32 nodes. Each node has 8 CPU cores at 2.6 GHz, 24 GB memory, 146 GB storage and Gigabit Ethernet connection. The measured latency on the network is 0.227 ms on average; and (ii) FutureGrid Federation site: The FutureGrid federation site is deployed on a cloud infrastructure based on OpenStack. In particular, we have used the infrastructure located at San Diego Supercomputer Center. We have used instances of type medium, where each instance has 2 cores and 4 GB of memory, and small, where each instance has 1 core and 2 GB of memory. The networking infrastructure is DDR Infiniband and the measured latency of the cloud virtual network is 0.706 ms on average.

We ran the distributed NMF algorithm from Section 4.2 on the JHU dataset. The rank of the factorization was set to 9. The learnt features are shown in Figure 2.

6 Discussion and future extensions

Overcoming the "small N" problem is key to making breakthroughs in medical research, particularly in elucidating the etiology of complex conditions such as mental health disorders. There is a disconnect between contemporary approaches to healthcare research and the rapid advances made in computing, data management, and machine learning. Researchers cannot become expert in these technologies, so it is imperative that we design platforms and interfaces that allow them to leverage additional data and powerful computational methods to process them in a transparent and automated manner. In this paper, we presented a platform for multi-site collaboration using distributed machine learning algorithms with differential-privacy guarantees. In particular, we showed a distributed version of the orthogonal NMF algorithm applied to sMRI data spread across different hospitals (sites).

The $\mathbf{C^3}$ platform is a step towards realizing that future. By augmenting this framework with additional functionalities, privacy protections, and ability to easily access distributed datasets, we can provide researchers in a variety of settings the tools they need to uncover the causes and cures for complex diseases.

Differentially private distributed algorithms: One important extension of this framework is to incorporate metrics for privacy-preserving data processing. In particular, designing differentially private [6, 7] updates would allow a more rigorous quantification of the privacy risk. Consider an algorithm M that takes the data $\mathbf{X}$ as one input and outputs a result $\mathsf{M}(\mathbf{X})$. In our NMF update rule we have two such algorithms: $\mathsf{M}_\mathsf{S}(\mathbf{X}; \mathbf{A}, \mathbf{S})$ that updates $\mathbf{S}$ in (5) and $\mathsf{M}_\mathsf{A}(\mathbf{X}; \mathbf{A}, \mathbf{S})$ that updates $\mathbf{A}$ in (6). Differential privacy is a property of *randomized* algorithms—the randomization introduces uncertainty in the output that provides privacy protections. The goal of differential privacy is to mask the presence or absence of a particular individual's data in the data set. Let $\mathbf{X}' \in \mathbb{R}^{d \times N}$ be a data set in which $N - 1$ columns are identical to the data $\mathbf{X}$ and one column is different. We call any such $\mathbf{X}'$ a neighbor of the data matrix $\mathbf{X}$.

An algorithm M guarantees (ϵ, δ)-differential privacy if for any pair $(\mathbf{X}, \mathbf{X}')$ of data matrices that are neighbors,

$$\mathbb{P}\left(\mathsf{M}(\mathbf{X}) \in \mathcal{T}\right) \leq e^{\epsilon}\mathbb{P}\left(\mathsf{M}(\mathbf{X}') \in \mathcal{T}\right) + \delta. \tag{10}$$

for any set $\mathcal{T}$ of outcomes. That is, the chance that M produces an output in $\mathcal{T}$ is similar for both $\mathbf{X}$ and $\mathbf{X}'$.

In our system, we consider a scenario where the individual sites trust the central server but they do not trust each other. This means we will not consider differentially private versions $\mathsf{M}_\mathsf{S}(\mathbf{X}_k; \mathbf{A}, \mathbf{S}_k)$ of the update (7), but instead focus on differentially private methods $\mathsf{M}_\mathsf{A}(\mathbf{X}_k; \mathbf{A}, \mathbf{S}_k)$ for the Reduce step of the distributed NMF algorithm. Data-independent post-processing the output of a differentially private algorithm cannot degrade the privacy guarantee, so we our approach is to make a differentially private computation of the matrix $\mathbf{C}$ in (8). This is a simple sum of matrices, each of which is based on the private data of the local sites. The challenge here is to analyze the sensitivity [7] of the individual terms in the sum in terms of a change in the data at one site from $\mathbf{X}_k$ to $\mathbf{X}'_k$. Over the course of the iterations, the entries in the matrices $\mathbf{X}_k\mathbf{S}_k^\top$ may have a larger range than the original data. The key to this analysis is developing a perturbation analysis of distributed NMF; we leave this for future work.

Differentially private methods for machine learning are still under active research development [9, 10], and designing practical methods for distributed systems can be quite challenging [8, 21]. Designing domain-specific algorithms can often yield better performance than off-the-shelf solutions, and experience from implementation may yield different differentially private aggregation rules [22]. One system-level implementation which has shown promise is GUPT [17]—incorporating design concepts from that system may improve performance under CometCloud.

Enabling a suite of algorithms: Distributed machine learning is an active field of research that includes work on Spark [24] and GraphLab [13]. Our approach is different from the previous ones in the sense we go beyond a single data-center and target federated cyber-infrastructure that is geographically distributed. We are currently experimenting with other machine learning algorithms such as LASSO, SVM and ICA. We are also working on extending the API to make the integration of novel machine learning algorithms in the distributed framework seamless.

Augmenting CometCloud with policies: CometCloud allows the definition of various types of users' requirements by means of policies. These policies translate these requirements into low level operations that determine how the algorithms are executed. For example, we could have a privacy budget policy that limits how much information can be leaked from every site at every iteration of the algorithm during communication. This policy guides the system to constrain which one of those sites can participate at each iteration depending on the available privacy budget. The budget is adjusted after every iteration.

References

[1] S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization—provably. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 145–162, New York, NY, USA, 2012. ACM. ISBN 9781-450-3124-5-5. doi: 10.1145/2213977.2213994.

[2] A. Bischoff-Grethe, B. Fischl, I. Ozyurt, S. Morris, G. G. Brown, C. Fennema-Notestine, C. P. Clark, M. W. Bondi, T. L. Jernigan, and the Human Morphometry BIRN. A technique for the deidentification of structural brain MR images. In *Human Brain Mapping*, Budapest, Hungary, 2004.

[3] S. Choi. Algorithms for orthogonal nonnegative matrix factorization. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1828–1832. IEEE World Congress on Computational Intelligence, June 2008. doi: 10.1109/IJCNN.2008.4634046.

[4] J. Diaz-Montes, Y. Xie, I. Rodero, J. Zola, Ganapathysubramanian, and M. Parashar. Federated computing for the masses—aggregating resources to tackle large-scale engineering problems. *IEEE Computing in Science and Engineering (CiSE) Magazine*, 16(4):62–72, 2014.

[5] J. Diaz-Montes, M. Zou, R. Singh, S. Tao, and M. Parashar. Data-driven workflows in multi-cloud marketplaces. In *IEEE Cloud 2014*, 2014.

[6] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *Advances in Cryptology—EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503, Berlin, Heidelberg, 2006. Springer-Verlag. doi: 10.1007/11761679_29.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284, Berlin, Heidelberg, March 4–7 2006. Springer. doi: 10.1007/11681878_14.

[8] A. Haeberlen, B. C. Pierce, and A. Narayan. Differential privacy under fire. In *Proceedings of the 20th USENIX Conference on Security*, Berkeley, CA, USA, 2011. USENIX Association.

[9] Z. Huang, S. Mitra, and N. Vaidya. Differentially private distributed optimization. Technical Report arXiv:1401.2596 [cs.CR], arXiv, January 2014. URL http://arxiv.org/abs/1401.2596.

[10] Z. Ji, X. Jiang, S. Wang, L. Xiong, , and L. Ohno-Machado. Differentially private distributed logistic regression using private and public data. *BMC Medical Genomics*, 7(Suppl 1):S14, May 2014. doi: 10.1186/1755-8794-7-S1-S14.

[11] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401 (6755):788–791, Oct. 1999. ISSN 0028-0836.

[12] Z. Li and M. Parashar. Comet: A scalable coordination space for decentralized distributed environments. In *2nd Intl. Workshop on Hot Topics in Peer-to-Peer Systems*, 2005.

[13] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein. Distributed graphlab: A framework for machine learning and data mining in the cloud. *Proc. VLDB Endow.*, 5(8):716–727, Apr. 2012. ISSN 2150-8097. doi: 10.14778/2212351.2212354.

[14] A. L. McGuire, M. Basford, L. G. Dressler, S. M. Fullerton, B. A. Koenig, R. Li, C. A. McCarty, E. Ramos, M. E. Smith, C. P. Somkin, C. Waudby, W. A. Wolf, and E. W. Clayton. Ethical and practical challenges of sharing data from genome-wide association studies: The eMERGE Consortium experience. *Genome Research*, 21(7):1001–1007, July 2011. doi: 10.1101/gr.120329.111.

[15] S. A. Meda, N. R. Giuliani, V. D. Calhoun, K. Jagannathan, D. J. Schretlen, A. Pulver, N. Cascella, M. Keshavan, W. Kates, R. Buchanan, et al. A large scale ($n = 400$) investigation of gray matter differences in schizophrenia using optimized voxel-based morphometry. *Schizophrenia research*, 101(1):95–105, 2008.

[16] M. Mennes, B. B. Biswal, F. X. Castellanos, and M. P. Milham. Making data sharing work: The FCP/INDI experience. *NeuroImage*, 82(0):683–691, 2013. ISSN 1053-8119. doi: 10.1016/j.neuroimage.2012.10.064.

[17] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. GUPT: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 349–360, New York, NY, USA, 2012. ACM. doi: 10.1145/2213836.2213876.

[18] G. Pearlson. Multisite collaborations and large databases in psychiatric neuroimaging: Advantages, problems, and challenges. *Schizophrenia Bulletin*, 35(1):1–2, 2009. doi: 10.1093/schbul/sbn166.

[19] R. A. Poldrack, D. M. Barch, J. P. Mitchell, T. D. Wager, A. D. Wagner, J. T. Devlin, C. Cumba, O. Koyejo, and M. P. Milham. Toward open sharing of task-based fMRI data: the OpenfMRI project. *Frontiers in Neuroinformatics*, 7(12), July 2013. doi: 10.3389/fninf.2013.00012.

[20] J.-B. Poline, J. L. Breeze, S. Ghosh, K. Gorgolewski, Y. O. Halchenko, M. Hanke, C. Haselgrove, K. G. Helmer, D. B. Keator, D. S. Marcus, R. A. Poldrack, Y. Schwartz, J. Ashburner, and D. N. Kennedy. Data sharing in neuroimaging research. *Frontiers in Neuroinformatics*, 6(9), April 2012. doi: 10.3389/fninf.2012.00009.

[21] I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel. Airavat: Security and privacy for MapReduce. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation (NDSI)*, 2010.

[22] A. D. Sarwate, S. M. Plis, J. A. Turner, M. R. Arbabshirani, and V. D. Calhoun. Sharing privacy-sensitive access to neuroimaging and genetics data: a review and preliminary validation. *Frontiers in Neuroinformatics*, 8(35), 2014. doi: 10.3389/fninf.2014.00035.

[23] T. Sherif, P. Rioux, M.-E. Rousseau, N. Kassis, N. Beck, R. Adalat, S. Das, T. Glatard, and A. C. Evans. CBRAIN: a web-based, distributed computing platform for collaborative neuroimaging research. *Frontiers in Neuroinformatics*, 8(54), May 2014. doi: 10.3389/fninf.2014.00054.

[24] E. R. Sparks, A. Talwalkar, V. Smith, J. Kottalam, X. Pan, J. Gonzalez, M. J. Franklin, M. I. Jordan, and T. Kraska. MLI: An API for distributed machine learning. In *2013 IEEE 13th International Conference on Data Mining (ICDM)*, pages 1187–1192. IEEE, 2013.

[25] M. Wolfson, S. Wallace, N. Masca, G. Rowe, N. Sheehan, V. Ferretti, P. Laflame, M. Tobin, J. Macleod, J. Little, I. Fortier, B. Knoppers, and P. Burton. DataSHIELD: resolving a conflict in contemporary bioscience—performing a pooled analysis of individual-level data without sharing the data. *International Journal of Epidemiology*, 39:1372–1382, 2010. doi: 10.1093/ije/dyq111.

[26] B. T. Wyman, D. J. Harvey, K. Crawford, M. A. Bernstein, O. Carmichael, P. E. Cole, P. K. Crane, C. DeCarli, N. C. Fox, J. L. Gunter, D. Hill, R. J. Killiany, C. Pachai, A. J. Schwarz, N. Schuff, M. L. Senjem, J. Suhy, P. M. Thompson, M. Weiner, C. R. Jack, J. Kaye, J. Quinn, B. Lind, R. Carter, S. Dolen, L. S. Schneider, S. Pawluczyk, M. Beccera, B. M. Spann, J. Brewer, H. Vanderswag, A. Fleisher, J. L. Heidebrink, J. L. Lord, R. Petersen, S. S. Mason, C. S. Albers, D. Knopman, K. Johnson, R. S. Doody, J. Villanueva-Meyer, M. Chowdhury, S. Rountree, M. Dang, Y. Stern, L. S. Honig, K. L. Bell, B. Ances, J. C. Morris, M. Carroll, S. Leon, E. Householder, M. A. Mintun, S. Schneider, A. Oliver, D. Marson, R. Griffith, D. Clark, D. Geldmacher, J. Brockington, E. Roberson, H. Grossman, E. Mitsis, L. deToledo Morrell, R. C. Shah, R. Duara, D. Varon, M. T. Greig, P. Roberts, M. Albert, C. Onyike, D. D'Agostino, S. Kielb, J. E. Galvin, D. M. Pogorelec, B. Cerbone, C. A. Michel, H. Rusinek, M. J. de Leon, L. Glodzik, S. De Santi, P. Doraiswamy, J. R. Petrella, T. Z. Wong, S. E. Arnold, J. H. Karlawish, D. Wolk, C. D. Smith, G. Jicha, P. Hardy, P. Sinha, E. Oates, G. Conrad, O. L. Lopez, M. Oakley, D. M. Simpson, A. P. Porsteinsson, B. S. Goldstein, K. Martin, K. M. Makino, M. Ismail, C. Brand, R. A. Mulnard, G. Thai, C. Mc-Adams-Ortiz, K. Womack, D. Mathews, M. Quiceno, R. Diaz-Arrastia, R. King, M. Weiner, K. Martin-Cook, M. DeVous, A. I. Levey, J. J. Lah, J. S. Cellar, J. M. Burns, H. S. Anderson, R. H. Swerdlow, L. Apostolova, K. Tingus, E. Woo, D. H. Silverman, P. H. Lu, G. Bartzokis, N. R. Graff-Radford, F. Parfitt, T. Kendall, H. Johnson, M. R. Farlow, A. M. Hake, B. R. Matthews, S. Herring, C. Hunt, C. H. van Dyck, R. E. Carson, M. G. MacAvoy, H. Chertkow, H. Bergman, C. Hosein, S. Black, B. Stefanovic, C. Caldwell, G. Y. Hsiung, H. Feldman, B. Mudge, M. Assaly, A. Kertesz, J. Rogers, D. Trost, C. Bernick, D. Munic, D. Kerwin, M. M. Mesulam, K. Lipowski, C. K. Wu, N. Johnson, C. Sadowsky, W. Martinez, T. Villena, R. S. Turner, K. Johnson, B. Reynolds, R. A. Sperling, K. A. Johnson, G. Marshall, M. Frey, J. Yesavage, J. L. Taylor, B. Lane, A. Rosen, J. Tinklenberg, M. N. Sabbagh, C. M. Belden, S. A. Jacobson, S. A. Sirrel, N. Kowall, R. Killiany, A. E. Budson, A. Norbash, P. L. Johnson, T. O. Obisesan, S. Wolday, J. Allard, A. Lerner, P. Ogrocki, L. Hudson, E. Fletcher, O. Carmichael, J. Olichney, C. DeCarli, S. Kittur, M. Borrie, T. Y. Lee, R. Bartha, S. Johnson, S. Asthana, C. M. Carlsson, S. G. Potkin, A. Preda, D. Nguyen, P. Tariot, A. Fleisher, S. Reeder, V. Bates, H. Capote, M. Rainka, D. W. Scharre, M. Kataki, A. Adeli, E. A. Zimmerman, D. Celmins, A. D. Brown, G. D. Pearlson, K. Blank, K. Anderson, R. B. Santulli, T. J. Kitzmiller, E. S. Schwartz, K. M. Sink, J. D. Williamson, P. Garg, F. Watkins, B. R. Ott, H. Querfurth, G. Tremont, S. Salloway, P. Malloy, S. Correia, H. J. Rosen, B. L. Miller, J. Mintzer, K. Spicer, D. Bachman, S. Pasternak, I. Rachinsky, J. Rogers, A. Kertesz, D. Drost, N. Pomara, R. Hernando, A. Sarrael, S. K. Schultz, L. L. Ponto, H. Shim, K. E. Smith, N. Relkin, G. Chaing, L. Raudin, A. Smith, K. Fargher, and B. A. Raj. Standardization of analysis sets for reporting results from ADNI MRI data. *Alzheimers Dement*, 9(3):332–337, May 2013. doi: 10.1016/j.jalz.2012.06.004.