Towards General-purpose Infrastructure for Protecting Scientific Data Under Study

Andrew Trask
OpenMined
University of Oxford
andrew@openmined.org

Kritika Prakash
OpenMined
IIIT Hyderabad
kritika.prakash@research.iiit.ac.in

Abstract

The scientific method presents a key challenge to privacy because it requires many samples to support a claim. When samples are commercially valuable or privacy-sensitive enough, their owners have strong reasons to avoid releasing them for scientific study. Privacy techniques seek to mitigate this tension by enforcing limits on one's ability to use studied samples for secondary purposes. Recent work has begun combining these techniques into end-to-end systems for protecting data. In this work, we assemble the first such combination which is sufficient for a privacy-layman to use familiar tools to experiment over private data while the infrastructure automatically prohibits privacy leakage. We support this theoretical system with a prototype within the Syft privacy platform using the PyTorch framework.

The scientific method has become one of humanity's most successful and fundamental sources of truth and our chief weapon of innovation and prosperity [49, 47]. However, it has a core constraint, it relies on the ability to observe the subject we want to understand [2]. Despite the so-called "big data revolution", not all data is universally available [28]. Specifically, the more personal and/or valuable the hypothesis we wish to validate, the more personal and/or valuable the data it requires (i.e., if an output is sensitive or valuable so too is the input capable of creating it). This creates a dilemma for holders of particularly personal and valuable data; Not only does every customer instantly becomes a competitor for all uses of the data (even and especially within academia), but every act of sharing carries with it significant ethical and legal risk. [3, 28, 4, 27, 19]. These combined effects polarize the data marketplace such that data is either shared as rapidly as possible or not at all.

While the former consequence is perhaps more well known, the latter is particularly unfortunate. The inability to analyze society's most valuable and personal data is the inability to answer life's most valuable and personal questions - perhaps the questions most worth answering. In this work, we argue that the key to solving this market failure is the creation of an end-to-end system for the automatic protection of data under study - such that data can be studied without being shared and without data owners actively participating in experiments. We propose such a system and design an open source prototype.

Technical Guarantees for Protecting Data Under Analysis The central puzzle of privacy is that we need to share information to collaborate but we cannot limit our collaborators from using such information against us. Privacy enhancing technologies seek to allow information to be collaboratively used in a way that its future use can be carefully limited by the data owner. In the context of empirical research, we observe that these technologies provide three guarantees involving two personas: the

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

data scientist and the data owner. These three guarantees are: protecting data from being copied by the data scientist, preventing statistical queries/results from being copied by the data owner, and preventing such statistical techniques from memorizing data in a way that the original data could be later extracted from it. With some exceptions, privacy enhancing technologies can be neatly grouped into these three categories when used in the context of empirical research.

Preventing a data scientist from copying and subsequently using data can be accomplished by bringing the algorithm to the data via federated learning/analytics [38, 12]. Preventing a data owner from copying and subsequently using the algorithm sent to the data can be accomplished using any flavor of encrypted computation: homomorphic encryption, secure multi-party computation, and functional encryption bogdanov2014input, barbosa2012delegatable. Preventing the statistical computation from memorizing data can be accomplished through output privacy, most notably differential privacy [21, 20, 40, 22, 1, 45]. If combined properly, a system with such a combination of integrated techniques could provide end-to-end guarantees sufficient for general data science over private data.

0.1 System Requirements

Despite tremendous progress on the theoretical capabilities of these systems, no implementation has yet emerged as a general-purpose alternative sufficient for scientists at large to no longer aggregate data (see supplementary material for a comprehensive overview of existing projects). We assert that such a system would require the following integrated features:

- **RPC (Federated Learning):** allow one to work with data on machines they do not control.
- **Arbitrary Pre-publish and Post-publish Differentail Privacy Composition:** facilitate efficient tracking of arbitrary remote computation both before (entity l2-norm) and after (composition) variables are made public [43, 55].
- **User-level Permissions:** require certain privacy budget constraints to be met before statistical results can be released to a data scientist user.
- **Adaptive Budgeting, Filter, and Approximate Odometer:** allow arbitrary exploration of the data while informing/limiting the data scientist based on how much budget remains [25].
- **Budget Simulations:** allow for remote analysis to occur which tracks a hypothetical budget so that a data scientist can measure whether, at the end of many compositions, the accuracy gained from their model would be worth the privacy budget spend and, if it does not, decide not to download any of the results (actually spend the budget).
- **Individual DP:** track privacy at the individual level for various reasons. Chief among them is the need for individuals represented in multiple datasets (perhaps even at multiple institutions) to ensure that there is an upper bound on the total amount of unique statistical information released about them. This is what actually prevents harm [23].

Nearly all of these properties exist in at least one tool or theoretical contribution. The exception is sufficiently automatic sensitivity. Our first major contribution is a general pre-publish composition language (flexible enough for any polynomial function) followed by a proposal for how it can be combined with previously proposed components to create an end-to-end system with these attributes. We finish with empirical baselines measuring the performance of a prototype system.

1 PrivateScalar

We propose a novel tool for sensitivity analysis which models a database query as a polynomial over scalar values. Each free variable corresponds to an input variable contributed from a unique entity.

Definition 1.1 (PrivateScalar) *Let y be a private scalar constructed using inputs from n entities formed with the following metadata (Private variables are **bold**):*

y^g : *($\mathbb{R}^n \rightarrow \mathbb{R}$) a polynomial function with a single, clipped indeterminate for each entity contributing to $\boldsymbol{y}$.*

$\boldsymbol{y}^x$: *the vector $\boldsymbol{y}^x \in \mathbb{R}^n$ represents the underlying value of each indeterminate within y^g which, if input to the polynomial, returns $\boldsymbol{y}^g(\boldsymbol{y}^x) = \boldsymbol{y}$.*

y^f *: (floor) each element of the vector* $y^f \in \mathbb{R}^n$ *represents the minimum possible value of the indeterminate* y^x_i *input into* y^g*.*

y^c *: (ceiling) each element of the vector* $y^c \in \mathbb{R}^n$ *represents the maximum possible value of the indeterminate* y^x_i *input into* y^g*.*

y *: the value of the private scalar, taken by clipping each indeterminate* $\mathbf{y}^x$ *within the range of* y^c *and* y^f *and passing it into the polynomial* y^g

Let us consider an example. Consider 100 individuals each contributing their age to a study. Entity i would initialize a PrivateScalar with an internal polynomial with 100 indeterminates, all with factors equal to 0 except for the i^{th} factor which is 1. Similarly, y^x would also be a one-hot vector, with entity i's age represented in the i^{th} position. Executing $\mathbf{y}^g(\mathbf{y}^x)$ would thus simply return entity i's age. However, if one wished to compute any arbitrary function over the 100 ages (such as a sum, mean, or arbitrary polynomial), each operation would manipulate the polynomial over inputs instead of manipulating the inputs themselves. This keeps the inputs disentangled as operations construct a complex query, the result of which is only calculated when the result is to be published (with noise).

We now consider how PrivateScalar can be applied within the context of the individual Rényi DP of [25]. Example 1.2 from [25] shows how to determine the entity-specific epsilon spend per query in the context of a Lipschitz function over a database (See appendix or [25] for the definition of individual RDP using notation matching this example).

Example 1.2 (Lipschitz analyses) *Suppose that* $g : (\mathbb{R}^d)^n \to \mathbb{R}^{d'}$ *is* L_i*-Lipschitz in coordinate* i*. For* $\phi\colon \mathcal{X} \to \mathbb{R}^d$*, let* $\mathcal{A}(S) = g(\phi(X_1), \dots, \phi(X_n)) + \xi$*,* $\xi \sim N(0, \sigma^2\mathbb{I}_{d'})$*. Assume that for some* $X^\star$*,* $\phi(X^\star)$ *is the origin. By using* $X^\star$ *to replace a removed element (namely* $S^{-i} = (X_1, \dots, X_{i-1}, X^\star, X_{i+1}, \dots, X_n)$*), then* $\mathcal{A}$ *satisfies* $\left(\alpha, \frac{\alpha L_i^2 \|\phi(X_i)\|_2^2}{2\sigma^2}\right)$ *individual RDP for* X_i*.*

Given that g is a query over private data X (where each element of X_i comes from unique entity i), the key question this example answers is: how much privacy budget does entity i spend when the output of $\mathcal{A}(S) = g(\phi(X_1), \dots, \phi(X_n)) + \xi$ is made public? The answer to this question is $\left(\alpha, \frac{\alpha L_i^2 \|\phi(X_i)\|_2^2}{2\sigma^2}\right)$, the two key terms of which are L_i^2 and $\|\phi(X_i)\|_2^2$, which PrivateScalar reveals. PrivateScalar mirrors this definition. g corresponds to y^g, an arbitrary polynomial over y^x, which corresponds directly to X. $\phi()$ is expressed within the factors of the polynomial. Thus, recovering the two key terms L_i^2 and $\|\phi(X_i)\|_2^2$ are as simple as considering the Lipschitz bound on each free variable in the polynomial and the L2 norm of the input respectively.

Recovering The Lipschitz Constant Any part of the query which involves information mixing (non additively) between multiple entities is captured in g. The term L_i^2 refers to the squared Lipschitz constant of g, with respect to the output of $\phi(X_i)$. This Lipschitz constant is not over an infinite range, however, but is only over the range of values expressed by the removal/replacement of $\phi(X_i)$.

However, while the L2 norm of the input is easy to compute, the Lipschitz bound on g can be more complex, because the derivative of y^g with respect to X_i may be conditioned on private data from entities other than i (if y^g multiplies two or more free variables). In the literature it is often assumed that each entity is able to see the remaining budget with respect to itself, thus each individual's budget should only be conditioned on private data of itself (and public data from others)[1]

To avoid this, challenge we instead consider the maximum possible derivative over all possible y^g polynomials given the publicly known ranges of all of its free variables (as defined by y^f and y^c, which are public). While computing this derivative can be challenging for complex polynomials, some special cases exist.

Special Cases In the case that the polynomial is comprised exclusively of non-negative factors and free variables, the polynomial becomes positively monotonic. In this case, finding the maximum possible derivative can be computed in closed form by considering y^g when all inputs y^x equal

[1] It is unclear whether this is a problem in all settings - as each entity need not necessarily know the current budget with respect to their data, and epsilon is considered private in [25]. We offer a conservative alternative for sake of generality.

exactly their upper bounds y^c. In the case that y^g is a first-degree polynomial, then the Lipschitz constant is equal to the public coefficient corresponding to y^x.

Thus, for any private scalar we seek to publish, which may have been formed from any polynomial over entity information, the amount of privacy spend for each contributing entity can be determined. This makes PrivateScalar comparable to previous work in sensitivity type systems, with several advantages over prior works (see Table 2 for prior works):

- in the spirit of [25], our sensitivity system employs some data-dependent information (namely x_i), which is tighter than previous purely data-independent approaches.
- this tool is capable of private-private multiplication between, and non-linear functions over, private values - which (to our knowledge) no existing tool for sensitivity analysis can bound.

The primary constraint of the PrivateScalar data-structure is one of performance; if one performs many computation on PrivateScalar values which involve multiple entities and potentially negative values, the underlying polynomial is likely to become very large, and the Lipschitz bounds expensive to compute.

2 An End-to-End System for Private Data Analysis

To fulfill the requirements of section 0.1, we combine PrivateScalar with the following tools toward an end-to-end system.

- **RPC (Federated Learning):** we leverage the RPC federated learning capabilities of the PySyft PPML framework.
- **Arbitrary Pre-publish and Post-publish Composition:** pre-publish composition is accomplished via PrivateScalar, and post-publish composition via the autodp tool of [55].
- **Permissions:** We integrate with the new 0.3.0 alpha release of PySyft, whose RPC framework has object-level, user-level permissions.
- **Adaptive DP Filter and Approximate Odometer:** we adopt the approach of [25].
- **Budget Simulations:** a data scientist can also copy their current odometer (remaining privacy budget) into a simulated data structure, then using this copy to simulate how a budget could be sent by telling this simulated odometer that it is publishing objects that haven't yet actually been downloaded. In this way, a data scientist can plan how a budget could be spent and search for the optimal privacy/accuracy tradeoff for their system before actually spending the true budget (downloading teh results).
- **Individual DP:** we augment the autodp framework of [55] with the individual DP method proposed in [25] within the PySyft framework.

While length will not allow a full exposition of each of these features, perhaps the most important integration to discuss is that between the permissions system of PySyft and the privacy budgeting mechanisms proposed above. Importantly, this means that a data owner can set a privacy budget for a data scientist, and the data scientist can perform any analysis they desire (and download results) as long as they stay under their budget. The combination between PrivateScalar and autodp within an existing statistical tool (PySyft + PyTorch) is, we argue, an important threshold in facilitating arbitrary data science over private data such that the data owner may rely heavily on automation to protect their information.

3 Conclusion and Future Work

In this work we survey recent techniques to propose a system sufficient to achieve an important milestone for the broader scientific community: the ability for a non-technical data owner to allow a data scientist to safely perform arbitrary data analysis over their private information such that the infrastructure can automatically protect the data under study (where "protect" is defined by a privacy budget). While most components for our proposed system exist in recent work, we identify and propose an important missing piece, online, arbitrary sensitivity (pre-publish) composition, and propose a tool capable of satisfying it. Finally, we argue that when combined with previous work in the way we recommend, the protection of private data while under (somewhat) arbitrary data analysis crosses an important usability threshold: automation. We present our prototype implementation as an open-source tool for further maturation[2]. Future work will focus on shoring up side-channel attacks, increasing computational performance, tightening differential privacy bounds, extending PrivateScalar beyond polynomials to threshold functions, and integrating more closely with other encrypted computation techniques offered by PySyft.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] Hanne Andersen and Brian Hepburn. Scientific Method. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2016 edition, 2016.

[3] Giorgio A Ascoli. Sharing neuron data: carrots, sticks, and digital records. *PLoS Biol*, 13(10):e1002275, 2015.

[4] Giorgio A Ascoli, Patricia Maraver, Sumit Nanda, Sridevi Polavaram, and Rubén Armañanzas. Win–win data sharing in neuroscience. *Nature methods*, 14(2):112–116, 2017.

[5] Arthur Azevedo de Amorim, Marco Gaboardi, Justin Hsu, and Shin-ya Katsumata. Metric semantics for probabilistic relational reasoning, 07 2018.

[6] Manuel Barbosa and Pooya Farshim. Delegatable homomorphic encryption with applications to secure outsourcing of computation. In *Cryptographers' Track at the RSA Conference*, pages 296–312. Springer, 2012.

[7] Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. Higher-order approximate relational refinement types for mechanism design and differential privacy. *ACM SIGPLAN Notices*, 50(1):55–68, 2015.

[8] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[9] Sarah Bird, Joshua Allen, and Kathleen Walker. Whitenoise: A platform for differential privacy. 2020.

[10] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the Symposium on Operating Systems Principles (SOSP)*, pages 441–459, 2017.

[11] Dan Bogdanov, Peeter Laud, Sven Laur, and Pille Pullonen. From input private to universally composable secure multi-party computation primitives. In *2014 IEEE 27th Computer Security Foundations Symposium*, pages 184–198. IEEE, 2014.

[12] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *CoRR*, abs/1902.01046, 2019.

[13] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H. McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings, 12 2018.

[14] Hui Cao, Shubo Liu, Renfang Zhao, and Xingxing Xiong. Ifed: A novel federated learning framework for local differential privacy in power internet of things. *International Journal of Distributed Sensor Networks*, 16(5):1550147720919698, 2020.

[15] Hao Chen, Kim Laine, and Rachel Player. Simple encrypted arithmetic library-seal v2. 1. In *International Conference on Financial Cryptography and Data Security*, pages 3–18. Springer, 2017.

[2]Prototype will be shared after blind review is complete.

[16] Victor Costan and Srinivas Devadas. Intel sgx explained. *IACR Cryptol. ePrint Arch.*, 2016(86):1–118, 2016.

[17] Morten Dahl, Jason Mancuso, Yann Dupis, Ben Decoste, Morgan Giraud, Ian Livingstone, Justin Patriquin, and Gavin Uhma. Private machine learning in tensorflow using secure computation. *arXiv preprint arXiv:1810.08130*, 2018.

[18] Georgios Damaskinos, Rachid Guerraoui, Anne-Marie Kermarrec, Vlad Nitu, Rhicheek Patra, and Francois Taiani. Fleet: Online federated learning via staleness awareness and performance prediction. *arXiv preprint arXiv:2006.07273*, 2020.

[19] Kord Davis. *Ethics of Big Data: Balancing Risk and Innovation*. O'Reilly Media, Inc., 2012.

[20] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[21] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[22] Cynthia Dwork and Guy N Rothblum. Concentrated differential privacy. *arXiv preprint arXiv:1603.01887*, 2016.

[23] Hamid Ebadi, David Sands, and Gerardo Schneider. Differential privacy: Now it's getting personal. *Acm Sigplan Notices*, 50(1):69–81, 2015.

[24] 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*, CCS '14, page 1054–1067, New York, NY, USA, 2014. Association for Computing Machinery.

[25] Vitaly Feldman and Tijana Zrnic. Individual privacy accounting via a renyi filter. *arXiv preprint arXiv:2008.11193*, 2020.

[26] Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C Pierce. Linear dependent types for differential privacy. In *Proceedings of the 40th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 357–370, 2013.

[27] Michal S Gal and Oshrit Aviv. The competitive effects of the gdpr. *Journal of Competition Law & Economics*, 2019.

[28] Julie Gould. Data sharing: Why it doesn't happen. *Nature Jobs*, 2015.

[29] David Gunning, Awni Hannun, Mark Ibrahim, Brian Knott, Laurens van der Maaten, Vinicius Reis, Shubho Sengupta, Shobha Venkataraman, and Xing Zhou. Crypten: A new research tool for secure machine learning with pytorch, 2019.

[30] Andreas Haeberlen, Benjamin C Pierce, and Arjun Narayan. Differential privacy under fire. In *USENIX Security Symposium*, volume 33, 2011.

[31] Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[32] Alex Ingerman and Krzys Ostrowski. Introducing tensorflow federated. *blog.tensorflow.org*, 2019.

[33] Qinghe Jing, Dong Daxiang, and contributors. Paddlefl, 9 20189.

[34] Noah Johnson, Joseph P Near, and Dawn Song. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

[35] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[36] Elisabet Lobo-Vesga, Alejandro Russo, and Marco Gaboardi. A programming framework for differential privacy with accuracy concentration bounds. *Proceedings of the ACM on Programming Languages*, 2020.

[37] Heiko Ludwig, Nathalie Baracaldo, Gegi Thomas, Yi Zhou, Ali Anwar, Shashank Rajamoni, Yuya Ong, Jayaram Radhakrishnan, Ashish Verma, Mathieu Sinn, et al. Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987*, 2020.

[38] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[39] Frank D McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.

[40] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[41] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. Gupt: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 349–360, 2012.

[42] Arjun Narayan and Andreas Haeberlen. Djoin: Differentially private join queries over distributed databases. In *Presented as part of the 10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*, pages 149–162, 2012.

[43] Joseph P Near, David Darais, Chike Abuah, Tim Stevens, Pranav Gaddamadugu, Lun Wang, Neel Somani, Mu Zhang, Nikhil Sharma, Alex Shan, et al. Duet: an expressive higher-order language and linear type system for statically enforcing differential privacy. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–30, 2019.

[44] NVIDIA. Clara, 9 20189.

[45] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*, 2016.

[46] Davide Proserpio, Sharon Goldberg, and Frank McSherry. Calibrating data to sensitivity in private data analysis. *arXiv preprint arXiv:1203.3453*, 2012.

[47] Stathis Psillos. *Scientific realism: How science tracks truth*. Routledge, 2005.

[48] Jason Reed and Benjamin C Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In *Proceedings of the 15th ACM SIGPLAN international conference on Functional programming*, pages 157–168, 2010.

[49] Paolo Rossi. *Francis Bacon: from magic to science*. Routledge, 2013.

[50] Indrajit Roy, Srinath TV Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. Airavat: Security and privacy for mapreduce. In *NSDI*, volume 10, pages 297–312, 2010.

[51] T. Ryffel, Andrew Trask, M. Dahl, Bobby Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach. A generic framework for privacy preserving deep learning. *ArXiv*, abs/1811.04017, 2018.

[52] Sameer Wagh et al. New directions in efficient privacy-preserving machine learning. 2020.

[53] Christopher Waites and Rachel Cummings. Pyvacy: Towards practical differential privacy for deep learning. 2019.

[54] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1698–1707. IEEE, 2020.

[55] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

[56] Wenbin Wei. Welcome to fate's documentation. 2018.

[57] Max Wong, H Moster, and Bryce Lin. eggroll, 12 2018.

[58] Hengchu Zhang, Edo Roth, Andreas Haeberlen, Benjamin C Pierce, and Aaron Roth. Fuzzi: A three-level logic for differential privacy. *Proceedings of the ACM on Programming Languages*, 3(ICFP):1–28, 2019.

Appendix

4 Protect Data From Being Copied

The scientific method requires a scientist to collect empirical evidence to support their claim. However, every popular statistical software tool assumes that its user has copied the information they seek to study onto computational resources they control. The natural consequence of this assumption is that scientists are technically capable of using such data for any purpose they desire - including uses the original data owners my oppose. Their data is, both literally and figuratively, “out of their hands”.

Theory The solution is simple but cumbersome. Instead of aggregating data to a single location, a scientist should delegate their analysis to data owners to run on their own hardware, thus avoiding the need to obtain a copy of the data they study. A huge burst of research has ensued in this direction seeking to break algorithms into discrete parts which can be run by multiple parties in this way. Federated learning (FL) refers such work in a machine learning setting and federated analytics to statistics more broadly [38, 12].

Implementations While much progress has been made in studying how to efficiently break algorithms apart, and many systems for FL are being built, no implementation has yet emerged as a general-purpose alternative sufficient for scientists at large to no longer aggregate data.

To the knowledge of these authors, all systems for FL require a scientist to actively coordinate with each data owner on every experiment (or with the builders of their software)[35]. If they seek to ensure their data is safe, the data owner must, for each experiment, read the code of each experiment for themselves (which assumes the data owner has the time and expertise to do so)[35]. Current FL infrastructure is high-touch, high-trust, high-expertise, and primarily suited for enterprises or mobile applications looking to do the same experiment over a long period of time with parties who trust their intentions (or don’t really have a choice)[35].

5 Protect Statistical Models From Being Copied

Setting aside open problems in federated learning, even idyllic FL still requires each data scientist to divulge their statistical techniques (and their work-in-progress model) to the data owner for training[35]. While the chief concern of privacy technology is the protection of personal information, in order for the protections of federated learning to be viable in a competitive marketplace, scientists (whether academic or commercial) need some ability to prevent data owners from taking advantage of their access to the statistics being created. The field of encrypted computation promises to address this.

Theory Encrypted computation, often called “input privacy”, allows for multiple parties to jointly compute a function without revealing their respective inputs to each other [11]. Within the field of cryptography, this field is called Secure Multi-party Computation (SMPC), special cases of which include Homomorphic Encryption and Functional Encryption [6]. Secure hardware can also provide encrypted computation as an alterantive to SMPC [16]. While all of these options can offer Turing complete encrypted computation, the ideal technique (or combination) for any particular application varies based on the available compute, ram, and network infrastructure.

Implementations Recent work has produced a flurry of libraries and chips for general purpose encrypted computation, particularly integer-level encrypted computation which is the preferred variety for statistical computation[16, 15]. Several tools exist which augment existing data science tools to run in an encrypted state [29, 17]. Furthermore, the long-standing challenge of providing encrypted computation frameworks which are performant on non-trivial machine learning tasks has, in general, been accomplished, although encrypted CPU training is typically still 10x+ slower than plain-text CPU training [52]. Creating more performance algorithms and implementations is still a very active area of research [52].

However, existing implementations fall short in a way similar to federated learning. Namely, no framework yet exists which is capable of tracking the encrypted computation while it is happening such that a data owner can automatically prevent data from simply being copied and sent to the data scientist during the computation process. And because the computation is encrypted, it's even more challenging for a data owner to read and understand the code they are running on their sensitive data (they would need to be an encrypted computation expert, of which there are very few).

For both federated learning and encrypted computation implementations, the missing piece is a security policy (linked to a user permissions system) which can actively and automatically ensure that each user of a system doesn't ever learn too much about the underlying data.

5.1 Prevent Statistical Models from Memorizing Data

The dominant solution for preventing statistical models from memorizing data is *differential privacy*. Introduced as a privacy constraint around database queries (simple statistics), it has been generalized to offer similar protections over even the most complex statistical analysis - ensuring that statistical results don't compromise the privacy of the records they describe.

Theory Proposed by [21, 20], differential privacy builds on the intuition that a query from a database is privacy preserving if removing or replacing any entry in the database doesn't change the result of the query. When a query's result does change given input perturbations, various "randomized algorithms" have been proposed to add noise to a database query in such a way that rigorous, worst-case bounds can be set on the probability that an input data-point could be inferred from the query result. Several popular modifications of the original DP definition have been proposed - for which many special-purpose mechanisms have been designed to find the best privacy/accuracy trade-off for specific algorithm types[40, 22, 1, 45].

Epsilon-delta DP Following the notation found in feldman2020individual , let $S = (X_1, \ldots, X_n)$ be an analyzed dataset, and $S^{-i} \overset{towards\text{def}}{=} (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$ be the analyzed dataset after removing point X_i. (ε, δ)-differential privacy (DP) measures privacy leakage using ε, an upper bound on the distance between queries on S and S^{-i}, and δ, the probability that the measure of distance fails.

Definition 5.1 *A randomized algorithm $\mathcal{A}$ is (ε, δ)-DP if for all datasets $S = (X_1, \ldots, X_n)$,*

$$\mathbb{P}\mathrm{r}\left[\mathcal{A}(S) \in E\right] \leq e^{\varepsilon}\, \mathbb{P}\mathrm{r}\left[\mathcal{A}(S^{-i}) \in E\right] + \delta, \text{ and } \mathbb{P}\mathrm{r}\left[\mathcal{A}(S^{-i}) \in E\right] \leq e^{\varepsilon}\, \mathbb{P}\mathrm{r}\left[\mathcal{A}(S) \in E\right] + \delta,$$

for all $i \in [n]$ and all measurable sets E [21, 20, 25].

Privacy Budgeting Each statistical query spends a certain degree of ε. In theory, the total amount of ε a data scientist is allowed to acquire in their analysis is called a *privacy budget*. If available, a data scientist can track their current spend of the budget using a *privacy odometer* for the exact value or a *privacy filter* to simply indicate whether or not the budget (or some partial threshold of it) has been exceeded.

Adaptivity and Scope However, the ability to measure a privacy budget by composing multiple rounds of privacy parameters ε and δ is very complex. The simplest and earliest composition theorems are non-adaptive and global, meaning that a data scientist must know all of the queries they want to run before viewing the output of the first one (no interactive exploring of the data), and the privacy budget refers to the leakage from the entire dataset as a whole (as opposed to individuals within the dataset whom we seek to protect). While the former might seem obviously problematic, the latter requires context.

Consider two data scientists, Bob and Alice, running their experiments against two medical data-sets at two different hospitals (respectively). Importantly, these data-sets have overlapping patients even though they're at different hospitals. If Bob and Alice are each given privacy budgets of $\varepsilon = 3$, if they compare their results they could (in theory) learn more than $\varepsilon = 3$ worth of information about the overlapping patients in their dataset!

That is to say, just limiting each scientist to a certain amount of ε budget doesn't ensure that people in the dataset are actually protected if scientists share results (or simply make them public). Instead, we should limit each *data subject* (in this case medical patients) to a certain amount of budget, no matter where that budget is being spent. DP which tracks a unique epsilon per individual is called individual DP.

Adaptive, Individual, Renyi-DP While a full survey of adaptive and individual DP methods is out of the scope of this work, we do focus on a particular extension of (ε, δ)-DP called Rényi DP-(RDP), which was recently extended with useful definitions and examples for adaptive composition with individual differential privacy by [25] and [23]. It is upon this work we will propose our end-to-end system.

5.2 Implementations

Recent work has seen many new tools for differential privacy, motivated by the desire for DP's complex analysis to be conveniently available both to practitioners and laymen alike. Works can be split into two groups by deployment environment: database differential privacy ([39], [46], [34], [36], [9]) and differential privacy within more general software programs: MapReduce ([50]), functional programs ([26, 30, 43]), federated datasets ([42]), and Python programs ([41, 55]).

Of the latter tools which support DP over arbitrary functions, there are two subgroups. Most tools exclusively focus on tracking and composing ε (post-query composition analysis), but some tools also track the arbitrary computation occurring before a publish event so that noise can be automatically calibrated to meet a certain budget [43, 48, 26, 5, 7, 58, 36]. Specifically, such tools track the "sensitivity" of a function's output to input perturbations. It is noteworthy, however, that some DP techniques calibrate noise based on measures other than sensitivity, but no automatic tools yet leverage them[25].

Relationship to Permissions Systems Similar to systems for federated learning and encryption, in our view the primary shortcoming in most systems for tracking DP is that nearly all systems capable of general purpose data science are decoupled from a remote-procedure call and object permissions system.

Put another way, while tools for analysis can tell you if your statistical analysis would leak private information if published, they can rarely use the conclusion of such analysis to explicitly prevent you from publishing anyway. The primary exception can be found within some database-query style DP tools, but these lack the ability to do arbitrary computation. A standout exception is the early work of [41] which does allow arbitrary programs and enforces a privacy budget against a data scientist adversary.

The Expressiveness of Automatic Sensitivity Tools However, while [41] does enforce a privacy budget for arbitrary programs, it does so by requiring the data analyst

While some hybrid sensitivity-composition tools exist the sensitivity analysis they offer is primarily linear and entity generic. To the best of the authors knowledge, a sensitivity analysis tool does not yet exist with supports analysis over the multiplication of two private values (without the result having infinite sensitivity) or a general purpose mechanism for tracking the sensitivity of non-linear functions. Additionally, no hybrid sensitivity-composition analysis tools are in a language commonly used for data science and by extension, none have been integrated into a popular statistical tool for general-purpose use.

Name	MoYr	Make	Base	RPC	OPC	AM	DP	PM	M	B	S	T
PySyft	Jul17	OpenMnd	TH	Y	Y	Y	3rd	Y	Y	Y	Y	Y
TFF	Sep18	Google	Any	N	N	N	3rd	N	N	N	Y	N
FATE	Sep18	WeBank	TH,+	N	N	N	3rd	N	N	N	Y	?
LEAF	Dec18	CMU	TF	N	N	N	3rd	N	N	N	Y	?
eggroll	Jul19	WeBank	TF, +	?	N	?	3rd	N	N	N	Y	?
PaddleFL	Sep19	Baidu	PD	Y	N	?	Y	N	N	N	Y	?
FLSim	Nov19	iQua	TH	N	N	N	N	N	N	N	Y	N
Clara	Dec19	NVIDIA	TF	N	N	Y	Y	N	N	N	Y	?
IBMFL	Jun20	IBM	KS+	N	N	N	Y	N	N	N	Y	Y
FLeet	Jun20	EPFL	TF?	N	N	?	Y	N	A	N	?	?
IFed	Jun20	WuhanU	CS	N	N	N	Y	N	N	N	Y	Y
FedML	Jul20	FedML	TH	Y	N	N	Y	N	A	N	Y	Y
Flower	Jul20	Cmbrdge	Any	Y	N	N	3rd	N	Y	?	Y	Y

Table 1: Federated Learning systems listed in order of publication (earlier of paper or Github repo). Name: the name of the system. MoYr: the month and year of publication. Make: the sponsoring organization. Base: the primary ML framework (TH=PyTorch, TF=Tensorflow, PD=Paddle, KS=Keras, CS=Custom, Any=Arbitrary, += multiple truncated for space). RPC: can a data scientist / coordinator node push jobs to workers or do workers pull them? OPC: Object level RPC - can a data scientist / coordinator interactively control arbitrary objects on the data nodes. AM: Can a data scientist / coordinator set/change the model architecture being trained without having access to client workers (or having to restart them)? DP: Does the framework natively support some kind of Differential Privacy (3rd = third party library can support). PM: Does the framework have a permissions system such that the data scientist/coordinator is considered a malicious adversary where the infrastructure's job is to prevent them from using their access to steal private data. M: Does the framework have mobile support (A=Android only)? B: Can the framework run in the browser? S: Can the framework run on servers? T: Can the framework run on IoT devices? [51, 32, 56, 13, 57, 33, 44, 37, 18, 14, 31, 8, 54]

Name	Based	RPC	ORPC	FSA	ISA	AC	IDP	RDP	PM	ML	UAPI
PINQ	-	SQL	No	Yes	No	API	No	No	Yes	No	C#
Airavat	-	MapR	No	No	No	Map	No	No	Yes	Yes	Java
Reed	-	Cust	No	Yes	Yes	?	No	No	No	Yes	Cust
Fuzz	PINQ	SQL	No	Yes	Yes	API	No	No	Yes	No	C#
GUPT	-	Pyth	No	Yes	No	Map	No	No	Yes	Yes	Pyth
wPINQ	PINQ	SQL	No	Yes	No	API	No	No	Yes	No	C#
DJoin	-	SQL	No	Yes	No	No	No	No	Yes	No	SQL
DFuzz	Fuzz	SQL	No	Yes	Yes	API	No	No	Yes	No	C#
RAPOR	-	No	No	Yes	No	API	No	No	No	No	C++
HOARe	DFuzz	SQL	No	Yes	Yes	API	No	No	Yes	Yes	C#
FLEX	-	SQL	No	Yes	No	No	No	No	Yes	No	SQL
Proclo	-	No	No	Yes	No	API	No	No	No	No	C++
Fuzzi	aRHL	No	No	Yes	Yes	API	No	Yes	No	Yes	Cust
Duet	-	No	No	Yes	Yes	?	No	Yes	No	No	Hskl
TFpriv	-	No	No	No	No	No	No	Yes	No	Yes	Pyth
pyvacy	-	No	No	No	No	No	No	Yes	No	Yes	Pyth
autodp	-	No	No	No	No	Yes	No	Yes	No	Yes	Pyth
WNoise	-	SQL	No	Yes	Yes	Yes	No	Yes	No	No	Pyth
GoogDP	-	No	No	No	No	No	No	No	No	No	R,Go
PyDP	GoDP	No	No	No	No	No	No	No	No	No	Pyth
SwftDP	GoDP	No	No	No	No	No	No	No	No	No	Swft
dp.js	GoDP	No	No	No	No	No	No	No	No	No	JS
JavaDP	GoDP	No	No	No	No	No	No	No	No	No	Java
ClojDP	GoDP	No	No	No	No	No	No	No	No	No	Cloj
diffPR	GoDP	No	No	No	No	No	No	No	No	No	R
Opacus	-	No	No	No	No	No	No	Yes	No	Yes	Pyth

Table 2: Differential privacy systems listed in order of publication year. Name:the name of the system. RPC: What language defines a query if the system allows remote operation. ORPC: Does the framework support object-level RPC (Yes) or just a static function API (No)? FSA: Can the system infer the sensitivity of supported functions (Yes) or does it need to be specified by the system designer (No)? ISA: Can FSA happen with knowledge of the underlying object sensitivity (i.e., is the sensitivity fixed per function (No) or dynamic to the object (Yes) over which the function is being called?). AC: How flexible is computation? (MAP: can pass in arbitrary map functions, but not reductions, API: arbitrary computation but only to the API the data owner explicitly develops, ?: Generally a flexible paradigm but unclear because some operations are limited or generate infinite budget spend). IDP: Supports individual differential privacy? RDP: Supports Rényi DP? PM: Integrated with a permissions system such that the user is considered an untrusted adversary who must stay under a privacy budget. ML: API flexible enough for general-purpose machine learning? UAPI: what language does the user use to query data?[39, 50, 48, 30, 41, 46, 42, 26, 7, 34, 58, 43, 55, 53, 1, 10, 24][4]