Online Algorithms with Limited Data Retention

Nicole Immorlica
Microsoft Research, Cambridge, MA, USA

Brendan Lucier
Microsoft Research, Cambridge, MA, USA

Markus Mobius
Microsoft Research, Cambridge, MA, USA

James Siderius
Tuck School of Business at Dartmouth, Hanover, NH, USA

Abstract

We introduce a model of online algorithms subject to strict constraints on data retention. An online learning algorithm encounters a stream of data points, one per round, generated by some stationary process. Crucially, each data point can request that it be removed from memory m rounds after it arrives. To model the impact of removal, we do not allow the algorithm to store any information or calculations between rounds other than a subset of the data points (subject to the retention constraints). At the conclusion of the stream, the algorithm answers a statistical query about the full dataset. We ask: what level of performance can be guaranteed as a function of m?

We illustrate this framework for multidimensional mean estimation and linear regression problems. We show it is possible to obtain an exponential improvement over a baseline algorithm that retains all data as long as possible. Specifically, we show that $m = \textsc{Poly}(d, \log(1/\epsilon))$ retention suffices to achieve mean squared error ϵ after observing $O(1/\epsilon)$ d-dimensional data points. This matches the error bound of the optimal, yet infeasible, algorithm that retains all data forever. We also show a nearly matching lower bound on the retention required to guarantee error ϵ. One implication of our results is that data retention laws are insufficient to guarantee the right to be forgotten even in a non-adversarial world in which firms merely strive to (approximately) optimize the performance of their algorithms. Our approach makes use of recent developments in the multidimensional random subset sum problem to simulate the progression of stochastic gradient descent under a model of adversarial noise, which may be of independent interest.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases online algorithms, machine learning, data, privacy, law

Digital Object Identifier 10.4230/LIPIcs.FORC.2024.10

Category Extended Abstract

Related Version *Full Version*: https://arxiv.org/abs/2404.10997

Funding *James Siderius*: This research was initiated while the author was a Research Intern at Microsoft Research.

Acknowledgements The authors thank Rad Niazadeh, Stefan Bucher, the Simons Institute for the Theory of Computing, participants at the CS and Law conference, seminar participants at the 2024 SIGecom Winter Meetings and the 2022 C3.ai DTI Workshop on Data, Learning, and Markets.

1 Introduction

Modern algorithms run on data. But as the potential uses for large datasets have expanded, so too have concerns about personal data being collected, retained, and used in perpetuity. Data protection laws are one response to these concerns, providing individuals the right to have their data removed from datasets. The EU's GDPR is a flagship example, encoding a "right

5th Symposium on Foundations of Responsible Computing (FORC 2024).
Editor: Guy N. Rothblum; Article No. 10; pp. 10:1–10:8
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

to be forgotten" and mandating compliance with data deletion requests [14]. Similar policies have taken effect across the United States, such as the California Consumer Privacy Act [5] and Virginia's Consumer Data Protection Act [6]. These policies specify rules governing deletion requests, but data removal is a complicated process. Data is not just stored: it is used to make decisions; it touches a vast array of metrics; it trains machine learning models. What, then, should it mean to remove data from a system? And how do such requests impact an algorithm's ability to learn?

A growing body of literature approaches these questions through the "outcome-based" lens of constraining the observed behavior and outcomes of an algorithm. For example, one might require that once a piece of data has been "removed" in response to a request, the algorithm's behavior should be indistinguishable (in a cryptographic sense) from one that does not have access to the data. Formalizing this idea leads to a myriad of details and modeling choices, and multiple notions of deletion-respecting algorithms have been proposed [4, 13, 7]. An alternative approach is to directly impose restrictions on an algorithm's internal implementation that regulate and define the data removal process. This "prescriptive" approach is especially appealing from a regulatory perspective, since such restrictions provide clear guidance on what is and is not allowed (and, by extension, what constitutes an enforceable violation). But on the other hand, the actual implications of any given implementation restriction are not necessarily clear *a priori*. Constraints that appear very restrictive at first glance may still allow undesirable behavior through clever algorithm design. This undesirable behavior may be exhibited even by non-adversarial firms that simply wish to optimize their performance. Thus, for any given definition of what is meant by an implementation that respects data deletion, it is crucial to explore the outcomes that are generated by optimal (or near-optimal) algorithm design.

In this paper we explore the latter approach. We consider a stark framework in which an online algorithm can retain *no state* beyond stored data, which is subject to deletion requests. We show that even under such a restriction and even for simple statistical tasks like mean estimation, an algorithm that can preemptively delete data from its dataset can effectively retain information about data that was supposedly removed while still following the letter of the law (i.e., limited data retention). Moreover, we show that one can use this flexibility to substantially improve performance on statistical tasks relative to naive baseline algorithms that follow the spirit of the law (i.e., the right to be forgotten).[1] These results suggest that even in a world where an algorithm can retain no internal state whatsoever beyond its dataset, the curation of the dataset itself can be used to encode substantial information about data that has been supposedly removed, and even non-adversarial designers who seek only to maximize performance may naturally develop algorithms that leak information that was requested to be deleted. These results emphasize the importance of laws that regulate outcomes as well as process.

1.1 A Framework for Limited Data Retention

We propose a framework for algorithm design built upon a literal interpretation of a request to remove data. A sequence of data points is observed by a learning algorithm that actively maintains a subset of the data that has been observed so far. Each data point can come with a request that it be stored for only m rounds, after which it must be removed from the

[1] For example, by retaining all data as long as is allowed by regulation and then using optimal statistical estimators on the retained dataset.

algorithm's subset.[2] We think of m as a legally-mandated period of time after which the algorithm is obligated to fulfill the request.[3] The algorithm is free to discard data earlier, if desired; the only constraint is that data cannot be retained beyond the m rounds.

Of course, removing data points from the "official" dataset has no bite without additional restrictions on what else the algorithm can store. To clarify the impact of removing data, we impose a crucial modeling assumption: the algorithm *cannot retain any state* between rounds other than the dataset itself. In other words, any statistics or intermediate calculations performed by the algorithm must be recomputed, when needed, using only the data currently in the dataset.[4]

Such an algorithm can be described by two procedures: one that maintains the dataset (i.e., given the current subset and an incoming data point, choose which subset to keep) and one that answers a query about the full data stream given the current subset, possibly employing some non-standard estimator tailored to the data retention strategy.

We initiate an exploration of this framework through the lens of two standard statistical tasks: mean estimation and linear regression. In the case of mean estimation, each data point is a drawn from an unknown distribution over $\mathbb{R}^d$ and the algorithm's goal is to recover the distribution's mean. In the case of linear regression, each data point is a pair (x, y) where x is a d-dimensional characteristic vector and y is generated through a linear function of x plus random noise, and the goal is to simulate the linear function on challenge queries. In each case, the mean squared error achievable by an estimator that can retain an entire data stream of T data points (without any requirement to remove data) improves linearly with T. We ask: what error is achievable by an algorithm that respects requests to remove incoming data points within m rounds?

One baseline algorithm is to simply retain all data as long as possible. That is, the algorithm retains all of the previous m data points, then returns the maximum likelihood estimator given the sample for the target query. This approach is equivalent to keeping a uniform subsample of m draws from the underlying distribution. For the mean estimation and linear regression tasks, a uniform subsample of size m yields an average squared error no better than $\Theta(1/m)$, even for draws from a Gaussian distribution. In other words, this baseline would need to retain data for $m = O(T)$ rounds to achieve error comparable to what is attainable from the entire data stream.

1.2 An Improved Data Retention Policy

We show that it is possible to achieve an exponential improvement relative to the baseline solution described above. We present an algorithm for mean estimation that achieves a loss guarantee comparable to the optimal estimator over all T data points, but that retains each data point for only $m = \textsc{Poly}(d, \log(T))$ rounds. In more detail, if m is at least $\Theta(d \log(d/\epsilon))$, then for any query time $T > Cd/\epsilon$ (where C is a constant depending on the input distribution) the expected squared error will be at most ϵ. For linear regression we achieve a similar guarantee, with $m = \Theta(d^2 \log(d) \log(d/\epsilon))$. Our algorithms are polytime: each update step takes time linear in d and $1/\epsilon$.

[2] All of our results extend directly to model where a removal request can be made in any round after the data arrives (not just at the moment of arrival), and the data must be removed within m rounds of the request.

[3] For example, under GDPR Article 12, any request to delete personal data must be honored "Without undue delay and in any event within one month of receipt of the request" [14].

[4] One can equivalently think of this as a policy describing which statistics can be kept between rounds; namely, those that could be directly recomputed using only the retained data.

We also present a nearly-matching lower bound: if $m = o\left(\frac{d\log(1/\epsilon)}{\log(d)\log\log(1/\epsilon)}\right)$ then the algorithm must have error greater than ϵ with constant probability, regardless of the output function used to map the final subsample to an estimate of the mean.

1.3 Related Work

There is a substantial line of literature that explores definitions of data removal, especially as it relates to data protection and privacy laws. The literature on machine unlearning, initiated by [4], explores the process of updated a trained machine learning model so that it cannot leak information about to-be-deleted data. This has led to a vast body of work exploring different definitions and designs; see [19, 23] for some recent surveys of this literature. Beyond machine learning contexts, a notion of data deletion in terms of not leaking information about the data and maintaining secrecy, termed deletion-as-confidentiality, was proposed by [13]. A more permissive notion that constrains the leakage of information only after a removal request, deletion-as-control, was explored by [7]. Such works employ outcome-based constraints on data leakage, often in combination with internal state restrictions. In contrast, we explore a prescriptive framework that directly restricts an algorithm's implementation and explore the extent to which these restrictions do (or do not) constrain the algorithm's achievable performance and observable outcomes. While our algorithm respects certain notions of random differential privacy [15], we show that simple implementation restrictions to delete data points is not sufficient to retain full differential privacy of the deleted data.

Our work is also related to a line of literature on non-uniform subsampling for linear regression. The typical goal is to draw a sample from a large (or infinite) pool of potential data items (x, y) to increase accuracy of resulting models. Early works employed leverage scores to weight the predictor vector x [10, 17]. This approach has been extended to other norms via low-distortion embeddings [18] and improved by including outcomes y via importance weighting [9, 24, 22]. In contrast, our approach is not based on independent sampling but rather adaptive sample maintenance with elements added and removed over time.

Our approach is also closely related to coreset construction [11], in which the goal is to develop a highly compressed summary of a large dataset that retains the ability to answer queries from a given query class. Effective constructions are known for many learning problems, including variations of regression for numerous risk functions [1]. In principle a coreset can retain additional information beyond an (unweighted) subset of the original data, whereas our framework motivates us to focus specifically on unweighted subsampling.

From a technical perspective, our constructions use online implementations of stochastic gradient descent (SGD), which itself makes heavy use of sampling [3]. Our algorithms effectively simulate the progression of SGD using subsamples to approximate estimates. These approximations introduce some poorly-controlled noise to the SGD process, which necessitates an analysis that is robust to adversarial noise; for this we provide a slight variation on an SGD analysis due to [21]. To show that small subsets of data suffice to approximate the evolution of a sequence of improving estimates of regression coefficients, we employ recent advances in the theory of the random subset sum problem (RSS) [2, 16, 8]. The application of the RSS problem in contexts where SGD is used has also been explored in literature related to the Strong Lottery Ticket Hypothesis (SLTH) in learning theory [12, 2, 20]. However, the application of RSS to our setting requires a novel analysis.

2 Techniques

The full version of our paper, which can be found at `https://arxiv.org/abs/2404.10997`, contains our formal model, theorem statements, and proofs. Here, we describe our techniques.

Our approach is to simulate the progression of stochastic gradient descent (SGD). Consider the mean estimation task, where the goal is to learn the distribution mean θ. As new data points arrive, a (non-subsample-based) SGD algorithm would maintain an estimate for θ; continuously updating the estimate in proportion to the noisy gradient estimate provided by fresh samples. Of course, such an SGD algorithm cannot be directly implemented as a subsampling algorithm because we are not allowed to directly maintain an estimate for θ, only subsets of data samples.

Our algorithm must therefore *simulate* the desired gradient steps: given a current estimator for θ (implied by the current subsample) and a proposed update (suggested by a gradient step), our algorithm will search recently-seen data points for a subset whose average approximates the target update. This approximation via a subset of data introduces noise into the gradient step. This noise is challenging to control, since the set of all averages of a subset of data points are heavily correlated with each other. We therefore treat this noise as adversarial, and note that SGD guarantees are robust to such adversarial noise as long as it is appropriately bounded. It turns out that having squared ℓ_2 distance of approximately ϵ between the target update and the closest subset-average would suffice to achieve our desired error rate.

How much memory is necessary to ensure that there is a subset of data points whose average is within ϵ of a given target point? As it turns out, recent developments in the random subset sum problem provides a surprising answer: for the single-dimensional case we only need that $m = \Omega(\log(1/\epsilon))$ and that the target point is not "too far" from the distributional average θ, where the asymptotic notation hides dependencies on parameters of the input distribution. To put this in perspective, this is asymptotically the same as what would be achievable if each of the 2^m subsets of samples were drawn independently of every other subset. To extend to d dimensions, we can either employ a multidimensional variant of the random subset sum problem, or target an error of ϵ/d on each dimension separately.

Extending this approach to linear regression tasks brings a new challenge. While SGD can still be used in this setting, the optimal estimator for the regression coefficients is not an empirical mean, so one cannot directly apply solutions to the random subset sum problem to encode an update step. For mean estimation, the natural estimator for θ is precisely an empirical mean of collected data points, so it is straightforward to encode an estimate of θ with a subsample that solves a random subset sum problem. For linear regression, the estimator for the coefficient vector is a specific transformation of a set of input (x, y) pairs, so we cannot directly apply the same trick. Instead, we will reduce to the mean estimation problem in a different way. Our proposed algorithm collects data points together into small groups that each generate an independent maximum likelihood estimate for the regression coefficients. As long as these groups are sufficiently large ($\Omega(d \log d)$ data points per group is enough) their corresponding estimates will be smoothly distributed near the true state. We can then think of these per-group maximum likelihood estimates as inputs to the random subset sum problem, and find a subset of them whose average approximates a proposed gradient step. This allows us to encode a gradient step by preserving the corresponding groups in our subsample.

To this point we have described ways to simulate gradient descent using subsampling. The algorithm effectively learns the desired statistics at the same rate as an algorithm with unbounded memory, and by maintaining the subsample judiciously it can encode what has been learned. One might naturally wonder at this point whether the memory requirements could be substantially improved with more clever encodings. Since we put no restriction on the mapping from subset to algorithm output, in principal an algorithm could use the retained subsample to encode complex features of the full data stream in some Byzentine manner, then decode this information at query time. While we do not rule this out, we show a lower bound: any algorithm that satisfies the recency property *requires* $m = \Omega((d/\log(d))(\log(1/\epsilon)/\log\log(1/\epsilon)))$ in order to achieve squared error ϵ, even for mean estimation. Roughly speaking, this bound follows because even if the algorithm succeeds in perfectly learning θ, it will not achieve error less than ϵ if for *every* subset of m data points, the output function applied to that subset falls outside the ϵ-ball centered at θ. For m smaller than our bound, we can take a union bound over all subsets to show that the probability of this bad event will be large no matter what output function is used.

3 Conclusions and Future Work

In this work we introduced a framework for online algorithms subject to strict data retention limits. The algorithms in our framework retain no state other than a subsample of the data, and each data point must be removed from the subsample after at most m rounds. We provide upper and lower bounds on the value of m needed to achieve error ϵ for mean estimation and linear regression. We find that it is possible to substantially outperform a naive maximal-storage baseline by adaptively and proactively curating the algorithm's dataset in order to improve its representativeness of the full data stream (including data that was to have been dropped).

Many technical questions are left open for future pursuits. We take a worst-case perspective that all data must be removed after m rounds, but one might consider a model where some data points can be retained for much longer. Does the presence of long-lived data alongside data that must be removed quickly enable different algorithmic approaches? Our subsampling framework can also be extended to other statistical tasks like non-linear regression, estimating higher moments, classification tasks, and so on. In each case, the algorithmic challenge is to dramatically reduce the size of a training set, online, so that a (perhaps specially-tailored) training process executed on the subsample can achieve performance approximately matching what is possible on the full data.

One could also apply our framework to non-stochastic or partially stochastic environments, where data is not necessarily generated according to a stationary process. Such environments can amplify the impact of individual data points (and their removal) on an algorithm's output and state. Of course, the achievable algorithmic guarantees might vary substantially depending on the assumptions made on the data. But even so, understanding the structure of optimal (or near optimal) algorithms can shed light on the manner in which algorithm designers may be incentivized to build systems in the face of data retention limitations.

Finally, one can explore whether alternative frameworks for data removal lead to different types of behavior in optimal algorithm designs. A step in this direction is to quantify the extent to which optimal algorithms in a given framework are "undesirable" from the perspective of data removal, and use this to directly compare frameworks. Such an endeavor can help to build a toolkit for building up algorithmic restrictions that align well with stated policy goals.

References

1 Olivier Bachem, Mario Lucic, and Andreas Krause. Practical coreset constructions for machine learning. *arXiv preprint arXiv:1703.06476*, 2017.

2 Luca Becchetti, Arthur Carvalho Walraven da Cuhna, Andrea Clementi, Francesco d'Amore, Hicham Lesfari, Emanuele Natale, and Luca Trevisan. On the multidimensional random subset sum problem. *arXiv preprint arXiv:2207.13944*, 2022.

3 Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer, 2010.

4 Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pages 463–480. IEEE, 2015.

5 California consumer privacy act. `https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5`, 2018. Cal. Civ. Code §§ 1798.100 et seq.

6 Consumer data protection act, 2021 h.b. 2307/2021 s.b. 1392. `https://lis.virginia.gov/cgi-bin/legp604.exe?ses=212&typ=bil&val=Hb2307`, 2021.

7 Aloni Cohen, Adam Smith, Marika Swanberg, and Prashant Nalini Vasudevan. Control, confidentiality, and the right to be forgotten. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 3358–3372, 2023.

8 Arthur da Cunha, Francesco d'Amore, Frédéric Giroire, Hicham Lesfari, Emanuele Natale, and Laurent Viennot. Revisiting the random subset sum problem. *arXiv preprint arXiv:2204.13929*, 2022.

9 Paramveer Dhillon, Yichao Lu, Dean P Foster, and Lyle Ungar. New subsampling algorithms for fast least squares regression. *Advances in neural information processing systems*, 26, 2013.

10 Petros Drineas, Malik Magdon-Ismail, Michael W Mahoney, and David P Woodruff. Fast approximation of matrix coherence and statistical leverage. *The Journal of Machine Learning Research*, 13(1):3475–3506, 2012.

11 Dan Feldman. Introduction to core-sets: an updated survey. *arXiv preprint arXiv:2011.09384*, 2020.

12 Damien Ferbach, Christos Tsirigotis, Gauthier Gidel, and Joey Bose. A general framework for proving the equivariant strong lottery ticket hypothesis. In *The Eleventh International Conference on Learning Representations*, 2022.

13 Sanjam Garg, Shafi Goldwasser, and Prashant Nalini Vasudevan. Formalizing data deletion in the context of the right to be forgotten. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 373–402. Springer, 2020.

14 General data protection regulation, 2016. Regulation 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), OJ 2016 L 119/1.

15 Rob Hall, Alessandro Rinaldo, and Larry Wasserman. Random differential privacy. *arXiv preprint arXiv:1112.2680*, 2011.

16 George S. Lueker. Exponentially small bounds on the expected optimum of the partition and subset sum problems. *Random Structures & Algorithms*, 12(1):51–62, 1998. `doi:10.1002/(SICI)1098-2418(199801)12:1<51::AID-RSA3>3.0.CO;2-S`.

17 Ping Ma, Michael Mahoney, and Bin Yu. A statistical perspective on algorithmic leveraging. In *International Conference on Machine Learning*, pages 91–99. PMLR, 2014.

18 Xiangrui Meng and Michael W Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 91–100, 2013.

19 Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*, 2022.

20 Ankit Pensia, Shashank Rajput, Alliot Nagle, Harit Vishwakarma, and Dimitris Papailiopoulos. Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient. *Advances in neural information processing systems*, 33:2599–2610, 2020.

21 Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1571–1578, 2012.

22 Daniel Ting and Eric Brochu. Optimal subsampling with influence functions. *Advances in neural information processing systems*, 31, 2018.

23 Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S Yu. Machine unlearning: A survey. *ACM Computing Surveys*, 56(1):1–36, 2023.

24 Rong Zhu. Gradient-based sampling: An adaptive importance sampling for least-squares. *Advances in neural information processing systems*, 29, 2016.