Reconstruction Attacks on Machine Unlearning: Simple Models are Vulnerable

Martin Bertran [*1] **Shuai Tang** [*1] **Michael Kearns** [1 2] **Jamie Morgenstern** [1 3] **Aaron Roth** [1 2]
Zhiwei Steven Wu [1 4]

Abstract

Machine unlearning is motivated by principles of data autonomy. The premise is that a person can request to have their data's influence removed from deployed models, and those models should be updated as if they were retrained without the person's data. We show that these updates expose individuals to high-accuracy *reconstruction attacks* which allow the attacker to recover their data in its entirety, even when the original models are so simple that privacy risk might not otherwise have been a concern. We show how to mount a near-perfect attack on the deleted data point from linear regression models. We then generalize our attack to other loss functions and architectures, and empirically demonstrate the effectiveness of our attacks across a wide range of datasets (capturing both tabular and image data). Our work highlights that privacy risk is significant even for extremely simple model classes when individuals can request deletion of their data from the model.

1. Introduction

As model training on personal data becomes commonplace, there has been a growing literature on data protection in machine learning (ML), which includes at least two thrusts:

Data Privacy Preserving the privacy of individual training examples is essential for ensuring privacy protection in ML applications. Privacy attacks that reveal privacy risks in trained models range from membership inference attacks (Shokri et al., 2017) at one extreme, which aim only to determine if a particular person's data was used in training, to reconstruction attacks (Dick et al., 2023) at the other extreme, which aim to recover the entire data records of many people. Differential privacy (Dwork et al., 2006) has emerged as the primary approach to ensure formal privacy, which provably defends against membership inference.

Machine Unlearning Individuals should have the flexibility to decide how their data is used even retroactively — including its influence on trained models. This has led to the field called *data deletion* or *machine unlearning* (Ginart et al., 2019; Cao & Yang, 2015). The idea is that, after an individual's data is deleted, the resulting model should be in the state it would have been had the model originally been trained without the individual in question's data. The primary focus of this literature has been on achieving or approximating this condition for complex models in ways that are more computationally efficient than full retraining ((Golatkar et al., 2020; Izzo et al., 2021; Gao et al., 2022; Neel et al., 2021; Bourtoule et al., 2021; Gupta et al., 2021).)

Practical work on both privacy attacks (like membership inference and reconstruction attacks) and machine unlearning has generally focused on large, complex models like deep neural networks. This is because (1) these models are the ones that are (perceived as) most susceptible to privacy attacks, since they have the greatest capacity to memorize data, and (2) they provide the most technically challenging case for machine unlearning (since for simple models, the baseline of just retraining the model is feasible). For simple (e.g., linear) models, the common wisdom has been that the risk of privacy attacks is low, and indeed, we verify in Appendix C that state-of-the-art membership inference attacks fail to achieve non-trivial performance when attacking linear models trained on tabular data. The main message of our paper is that the situation changes starkly when we consider privacy risks in the presence of machine unlearning. As we show, absent additional protections like differential privacy, requesting that your data be removed—even from a linear regression model—can expose you to a complete reconstruction attack. Informally, this is because it gives the adversary *two models* that differ in whether your data was used in training, which allows them to attempt a differencing attack. Additional related work are in B.

Our Contributions We consider the following threat model: An attacker has access to the parameters of some model both before and after a deletion request is made. The attacker also has sampling access to the data distribution, but no access to the training set of the models. In this setting, we first study

[*]Equal contribution [1]Amazon AWS AI/ML [2]University of Pennsylvania [3]University of Washington [4]Carnegie Mellon University. Correspondence to: Martin Bertran <maberlop@amazon.com>, Shuai Tang <shuat@amazon.com>.

exact reconstruction attacks on linear regression models. We give an attack that accurately recovers the deleted sample given the pair of linear models before and after sample deletion. This is made possible by leveraging the closed-form single-sample training algorithm for linear regression as well as the ability to accurately estimate the covariance matrix of the training data from a modestly sized disjoint sample of public data drawn from the same distribution.

We then extend our attack to the setting where the model consists of a (fixed and known) embedding function, followed by a linear layer. Our goal remains to recover the original data point (not just its embedding). This is a natural class of simple models on its own, and captures last-layer fine-tuning on top of a pre-trained model, which is common in which machine unlearning guarantees are offered.

Finally, we give a second-order unlearning approximation via Newton's method which extends our attack to generic loss functions and model architectures. This provides a way to approximate the gradient of a deleted sample with respect to the model parameters, and later the sample itself. We remark that Newton's update approximation has itself been proposed as a way to approximate unlearning (Izzo et al., 2021; Gao et al., 2022), and is naturally related to the literature on influence functions (Hampel, 1974; Koh & Liang, 2017; Zhang & Zhang, 2022), which examines the effect any (set of) samples has on the final trained model.

We experimentally demonstrate the effectiveness of our attack on a variety of simple tabular and image classification and regression tasks. The success of our attack highlights the privacy risks of retraining models to remove the influence of individual's data, without additional protections like differential privacy (as advocated by e.g. (Chourasia et al., 2022) in another context). Figure 1 shows several deleted samples from a model trained on CIFAR10 (Krizhevsky et al., 2009) alongside the recovered reconstructions for our method and a baseline based on public data. Experimental results can be found in Appendix D.

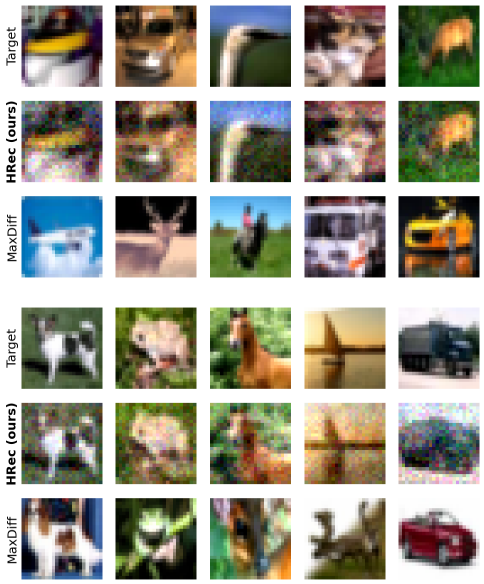


Figure 1: Visualization of samples reconstructed from a logistic regression model over a random Fourier feature embedding (4096) of the raw input, trained on CIFAR10 data. We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. HRec produces reconstructions similar to the deleted images both visually and quantitatively measured by cosine similarity.

2. Method

We first derive our attack for (regularized) linear regression models (with unknown regularization parameter), in which a model maintainer trains a linear regression model on the private training set $X_{\text{priv}} \in \mathbb{R}^{n\times d}, y_{\text{priv}} \in \mathbb{R}^n$, and then a user corresponding to a sample (x, y) asks to be removed from the training set. The model maintainer correspondingly trains a second model without this user's data ($X_{\text{priv}} \backslash x \in \mathbb{R}^{(n-1)\times d}, y_{\text{priv}} \backslash y \in \mathbb{R}^{n-1}$). The goal of our attack is to reconstruct the features of this deleted user, and our attack assumes access to the following items:

1. The linear regression model with parameters β^+ trained on $X_{\text{priv}}, y_{\text{priv}}$.
2. The updated linear regression model with parameters β^- trained on $X_{\text{priv}} \backslash x, y_{\text{priv}} \backslash y$
3. Public samples $X_{\text{pub}}, y_{\text{pub}}$ drawn from the same distribution as $X_{\text{priv}}, y_{\text{priv}}$

Here both models β^+, and β^- are the solution to a (regularized) linear regression problem. In particular,

$$\beta^+ = \arg\min_{\beta} \|X_{\text{priv}}\beta - y_{\text{priv}}\|_2^2 + \frac{\lambda}{2}\|\beta\|_2^2. \quad (1)$$

which has a closed-form solution as follows:

$$\beta^+ = C^{-1} X_{\text{priv}}^\top y_{\text{priv}}, \quad (2)$$

where $C = X_{\text{priv}}^\top X_{\text{priv}} + \lambda I$ is the (regularized) covariance matrix. Similarly, β^- can be written as

$$\beta^- = (C - xx^\top)^{-1}(X_{\text{priv}}^\top y_{\text{priv}} - x^\top y).$$

Note that the above formulation does not explicitly include an intercept term. Without loss of generality, the intercept can be learned by appending an additional $d + 1$st

"dummy feature" to each example x which always takes value $x_{d+1} = 1$. The intercept is now learned as an additional parameter β_{d+1} which multiplies the dummy feature.

2.1. Determining Direction of the Deleted Sample

We can express the updated parameter β^- in terms of β^+, C, x, y via the Sherman-Morrison formula:

$$\beta^+ = \beta^- + \frac{y - x^\top \beta^-}{1 + x^\top C^{-1} x} C^{-1} x. \tag{3}$$

By re-organizing terms, we have

$$C(\beta^+ - \beta^-) = \frac{y - x^\top \beta^-}{1 + x^\top C^{-1} x} x = \alpha(x, y) x, \tag{4}$$

where $\alpha(x, y)$ is a scalar function that depends on both x and y. It becomes immediate that the LHS $C(\beta^+ - \beta^-)$ only differs from the deleted sample x by a scalar α.[1]

Since we do not have access to the private training data X_{priv} or the regularization parameter λ, we use public samples X_{pub} to estimate the covariance matrix $\hat{C} = X_{\text{pub}}^\top X_{\text{pub}}$ without regularization. It remains to estimate the scalar $\alpha(x, y)$, which depends on both the deleted sample (x, y).

2.2. Determining Scale of Deleted Sample

We introduce an intercept normalization trick to determine the scale of the deleted sample.

Recall that we interpret the intercept as a parameter β_{d+1} that multiplies a dummy feature $x_{d+1} = 1$. We will use our knowledge that $x_{d+1} = 1$ to determine the scale of x. Making this dummy feature explicit, and assuming that $\hat{C}$ is our estimate of the covariance matrix excluding the dummy feature, we can write our previous analytical solution for our attack into the following expression that now explicitly accounts for the dummy feature:

$$\tilde{x} = \begin{bmatrix} \hat{C} & X_{\text{pub}}^\top \mathbf{1} \\ \mathbf{1}^\top X_{\text{pub}} & \mathbf{1}^\top \mathbf{1} \end{bmatrix} \left(\beta^+ - \beta^-\right) \in \mathbb{R}^{d+1}, \tag{5}$$

where $\mathbf{1} \in \mathbb{R}^n$ is the constant 1 vector. The reconstructed dummy feature is then

$$\tilde{x}_{d+1} = \begin{bmatrix} \mathbf{1}^\top X_{\text{pub}} & \mathbf{1}^\top \mathbf{1} \end{bmatrix} \left(\beta^+ - \beta^-\right) \in \mathbb{R} \tag{6}$$

Since we know that $x_{d+1} = 1$, we obtain the correct scaling factor by normalizing the reconstructed features in Eq. (5) by the scalar in Eq. (6). Practically, our attack only computes Eq. (5) once, and then normalizes features by its last dimension. Our attack is described in Algorithm 1.

[1] We note that samples that already satisfy $y = x^\top \beta^-$ make no impact on the updated model and thus are not recoverable.

3. Beyond Linear Regression

Our attack is derived for linear regression, which is a common simple workhorse model class. However, the same idea can be generalised beyond linear regression. The attack generalises immediately to any model which performs linear regression on top of a fixed embedding: our attack recovers the embedding of the deleted point, and reduces the problem to inverting the embedding. We can also generalise to other loss functions—The primary challenge is that we no longer have closed-form expressions for an "update", but we can approximate this.

3.1. Fixed Embedding Functions

Our attack is built upon the analytical update of adding or deleting a sample to a linear regression model, therefore, it also generalises to linear models trained on top of embeddings of the original features. Suppose that both parameter vectors β^+ and β^- along with the embedding function $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ are publicly known. Our attack first reconstructs the embeddings as in Eqs. (5) and (6), and then reconstructs features by finding a data point whose embedding best matches the reconstructed transformed features as in Eq. (8).

$$\tilde{z} = \begin{bmatrix} \phi(X_{\text{pub}})^\top \phi(X_{\text{pub}}) & \phi(X_{\text{pub}})^\top \mathbf{1} \\ \mathbf{1}^\top \phi(X_{\text{pub}}) & \mathbf{1}^\top \mathbf{1} \end{bmatrix} \left(\beta^+ - \beta^-\right), \tag{7}$$

$$\tilde{x} = \arg\min_x \left\| \frac{\tilde{z}}{\tilde{z}_{d'+1}} - \phi(x) \right\|. \tag{8}$$

Here, we assume that the embedding ϕ is fixed—that is, it doesn't change after deleting a sample. This is the case when e.g. performing last-layer fine-tuning on top of a pre-trained model, and for data-independent embeddings like random Fourier features.

3.2. Arbitrary Loss Functions

The analytical solution in Eq. (4) is derived in the context of linear models trained to minimize mean squared error. When optimizing other loss functions over other model classes, we no longer have closed-form solutions, but we can use Newton's method to approximate the "update function". Assume the model maintainer minimizes an empirical risk function as follows:[2]

$$\ell(\beta; X_{\text{priv}}, y_{\text{priv}}) = \frac{1}{n} \Sigma_{(x,y) \in \{X_{\text{priv}}, y_{\text{priv}}\}} \ell(\beta; x, y), \tag{9}$$

where $\beta^+ = \arg\min_\beta \ell(\beta; X_{\text{priv}}, y_{\text{priv}})$ and $\beta^- = \arg\min_\beta \ell(\beta; X_{\text{priv}} \setminus x, y_{\text{priv}} \setminus y)$ are the optimal model parameters before and after data deletion. We can use Newton's method (i.e. a 2nd-order Taylor approximation) to

[2] a regularization term $\lambda\Omega(\beta)$ can also be incorporated in the analysis but is here omitted for ease of presentation.

approximate the deletion step as

$$\beta^- \approx \beta^+ - H^{-1}\nabla\ell, \tag{10}$$
$$\text{where } H = \nabla^2_{\beta=\beta^+}\ell(\beta; X_{\text{priv}}\backslash x, y_{\text{priv}}\backslash y),$$
$$\nabla\ell = \nabla_{\beta=\beta^+}\ell(\beta; X_{\text{priv}}\backslash x, y_{\text{priv}}\backslash y), .$$

where $\nabla_{\beta=\beta^+}\ell$ refers to the derivative of ℓ with respect to β evaluated at $\beta = \beta^+$. By observing that $\beta^+ = \arg\min_\beta \ell(\beta; X_{\text{priv}}, y_{\text{priv}})$, we can use the first-order optimality conditions to get

$$\nabla_{\beta=\beta^+}\ell(\beta; X_{\text{priv}}, y_{\text{priv}}) = 0 \tag{11}$$
$$n\nabla_{\beta=\beta^+}\ell(\beta; X_{\text{priv}}\backslash x, y_{\text{priv}}\backslash y) = -\nabla_{\beta=\beta^+}\ell(\beta; x, y) \tag{12}$$

Intuitively, the gradient with respect to the parameter vector evaluated at β^+ on the remaining samples is equal to the negative of that on the deleted sample. Then, we have

$$n\nabla\ell = -\nabla_{\beta=\beta^+}\ell(\beta; x, y) \tag{13}$$

Replacing $\nabla\ell$ in Eq. (10) with Eq. (13), we get

$$\beta^- \approx \beta^+ + \frac{H^{-1}}{n}\nabla_{\beta=\beta^+}\ell(\beta; x, y), \tag{14}$$
$$nH(\beta^+ - \beta^-) \approx -\nabla_{\beta=\beta^+}\ell(\beta; x, y), \tag{15}$$

For linear regression, this approach is exact and recovers Eq (4). Compared to the special case of linear regression, in this generalisation, the Hessian of the loss function plays the role of the covariance matrix of the data in linear regression. In practice, we use the same public data trick to estimate the Hessian matrix H defined over the private data via m public samples drawn from the same distribution. That is:

$$\hat{H} = \frac{1}{m}\Sigma_{x',y'\in\{X_{\text{pub}}, y_{\text{pub}}\}}\nabla^2_{\beta=\beta^+}\ell(\beta; x', y'), \tag{16}$$

In general, our attack recovers the gradient of the loss at the deleted loss sample with respect to the model parameters. The situation simplifies when the model under attack is linear. Commonly used loss functions for classification tasks, including the cross-entropy loss and the hinge loss, produce gradient vectors that are proportional to data samples. For a linear model with a differentiable loss function, we have

$$\nabla_{\beta=\beta^+}\ell(\beta; x, y) = \frac{\partial\ell}{\partial(x^\top\beta)}\frac{\partial x^\top\beta}{\partial\beta}\Big|_{\beta=\beta^+}$$
$$= x^\top\frac{\partial\ell}{\partial(x^\top\beta)}\Big|_{\beta=\beta^+}, \tag{17}$$

Thus, the gradient for the deleted sample is proportional to the deleted sample. With the intercept normalization trick, we can recover the correct scaling factor. For non-differentiable loss functions, one can use smooth approximations to obtain gradients (Koh & Liang, 2017). Our goal is to expose the privacy risks of simple models, and hence, we keep our discussion focused on linear ones.

To attack models with an arbitrary loss function, we first estimate the Hessian on the public data $\hat{H}$, and then we obtain the gradient with respect to the deleted sample according to Eq. (15). We note that the Hessian matrix may be too large to represent in practice for larger models, since it scales quadratically in the number of model parameters. However, it is straightforward to use efficient implementations of Hessian-vector Product (HVP) in deep learning frameworks in Equation (15), and so there is no need to explicitly represent the Hessian. Other influence matrices can likewise be used in place of the Hessian, such as the Fisher information matrix described in (Ly et al., 2017; Bae et al., 2022; Teso et al., 2021).

Newton's update is a second-order Taylor approximation that is exact when it is applied to linear regression. When the loss function depends on higher-order moments, the approximation quality worsens, and the performance of our attack correspondingly degrades.

3.3. Multiclass Classification and Label Inference

The attack outlined in Eq. (15) provides an estimate of the gradient of the parameters with respect to the deleted sample $\nabla_{\beta=\beta^+}\ell(\beta; x, y)$. Recovering the sample from the parameter gradient can be done by using Eq. (8) in Section 3.1 and replacing the embedding function $\phi(x)$ with $\nabla_{\beta=\beta^+}\ell(\beta; x, y)$ in the inverse problem.

For models that make use of a single linear layer after a (potential) embedding function, we can simply recover the embedding directly using Eq. (17). Otherwise, we need to infer the deleted label y from the available information. This is not straightforward when there are multiple class-specific parameters in multi-class classification tasks.

Fortunately, most multi-class classification approaches make use of a softmax nonlinearity to output a probability vector over the space $\mathbb{P}_\mathcal{Y}$. As such, the derivative of the loss with respect to the bias parameter corresponding to the correct deleted label y should be negative, while the remaining bias terms should have a positive derivative for most standard loss functions. For example, the derivative of the bias terms of the softmax nonlinearity under cross-entropy loss is $\nabla_{b_j}[-\ln f_y(x; \beta^+)] = f_j(x; \beta^+) - \mathbf{1}[j = y]$. Where b_j denotes the bias term of the j-th class, and $f_j(x; \beta^+)$ denotes the probability assigned to label j by the model. Thus, we simply impute the deleted label to be

$$\hat{y} = \arg\min_j \nabla_{b_j}\ell(\beta; x, y)\big|_{\beta=\beta^+}. \tag{18}$$

With this label inference technique, our attack can also be generalized to attacking multi-class classification models.

References

Bae, J., Ng, N., Lo, A., Ghassemi, M., and Grosse, R. B. If influence functions are the answer, then what is the question? *Advances in Neural Information Processing Systems*, 35:17953–17967, 2022.

Balle, B., Cherubin, G., and Hayes, J. Reconstructing training data with informed adversaries. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1138–1156. IEEE, 2022.

Bertran, M., Tang, S., Kearns, M., Morgenstern, J., Roth, A., and Wu, Z. S. Scalable membership inference attacks via quantile regression. *arXiv preprint arXiv:2307.03694*, 2023.

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 141–159. IEEE, 2021.

Cao, Y. and Yang, J. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*, pp. 463–480. IEEE, 2015.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Carlini, N., Chien, S., Nasr, M., Song, S., Terzis, A., and Tramer, F. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914. IEEE, 2022.

Carlini, N., Hayes, J., Nasr, M., Jagielski, M., Sehwag, V., Tramer, F., Balle, B., Ippolito, D., and Wallace, E. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 5253–5270, 2023.

Chen, M., Zhang, Z., Wang, T., Backes, M., Humbert, M., and Zhang, Y. When machine unlearning jeopardizes privacy. In *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, pp. 896–911, 2021.

Chourasia, R., Shah, N., and Shokri, R. Forget unlearning: Towards true data-deletion in machine learning. *arXiv preprint arXiv:2210.08911*, 2022.

Dick, T., Dwork, C., Kearns, M., Liu, T., Roth, A., Vietri, G., and Wu, Z. S. Confidence-ranked reconstruction of census microdata from published statistics. *Proceedings of the National Academy of Sciences*, 120(8):e2218605120, 2023. doi: 10.1073/pnas.2218605120. URL https://www.pnas.org/doi/abs/10.1073/pnas.2218605120.

Ding, F., Hardt, M., Miller, J., and Schmidt, L. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

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

Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., and Ristenpart, T. Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing. In *23rd USENIX security symposium (USENIX Security 14)*, pp. 17–32, 2014.

Gao, J., Garg, S., Mahmoody, M., and Vasudevan, P. N. Deletion inference, reconstruction, and compliance in machine (un) learning. *arXiv preprint arXiv:2202.03460*, 2022.

Ginart, A., Guan, M., Valiant, G., and Zou, J. Y. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32, 2019.

Golatkar, A., Achille, A., and Soatto, S. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9304–9312, 2020.

Gupta, V., Jung, C., Neel, S., Roth, A., Sharifi-Malvajerdi, S., and Waites, C. Adaptive machine unlearning. *Advances in Neural Information Processing Systems*, 34: 16319–16330, 2021.

Hampel, F. R. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Izzo, Z., Smart, M. A., Chaudhuri, K., and Zou, J. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, pp. 2008–2016. PMLR, 2021.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ly, A., Marsman, M., Verhagen, J., Grasman, R. P., and Wagenmakers, E.-J. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

Neel, S., Roth, A., and Sharifi-Malvajerdi, S. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pp. 931–962. PMLR, 2021.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Salem, A., Bhattacharya, A., Backes, M., Fritz, M., and Zhang, Y. {Updates-Leak}: Data set inference and reconstruction attacks in online learning. In *29th USENIX security symposium (USENIX Security 20)*, pp. 1291–1308, 2020.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pp. 3–18. IEEE, 2017.

Teso, S., Bontempelli, A., Giunchiglia, F., and Passerini, A. Interactive label cleaning with example-based explanations. *Advances in Neural Information Processing Systems*, 34:12966–12977, 2021.

Wu, X., Fredrikson, M., Wu, W., Jha, S., and Naughton, J. F. Revisiting differentially private regression: Lessons from learning theory and their consequences. *arXiv preprint arXiv:1512.06388*, 2015.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zhang, R. and Zhang, S. Rethinking influence functions of neural networks in the over-parameterized regime. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 9082–9090, 2022.

Algorithm 1 Reconstruction Attack on Linear Regression

Inputs: Public Samples $X_{\text{pub}} \in \mathbb{R}^{m\times(d+1)}$, $y_{\text{pub}} \in \mathbb{R}^m$, Parameter Vectors β^+ and $\beta^- \in \mathbb{R}^{d+1}$
Output: Reconstructed Sample $\tilde{x}$
Estimate the covariance matrix $\hat{C} = X_{\text{pub}}^T X_{\text{pub}}$
Compute $\tilde{x}$ using Eq. (5)
Normalize $\tilde{x}$ using Eq. (6), $\tilde{x}_i = \tilde{x}_i/\tilde{x}_{d+1}, \forall i \in [1, d]$
Return first d dimensions of $\tilde{x}$

Algorithm 2 Generalized Attack

Required: Public Samples $X_{\text{pub}} \in \mathbb{R}^{m\times(d+1)}$, $y_{\text{pub}} \in \mathbb{R}^m$, Parameter Vectors β^+ and $\beta^- \in \mathbb{R}^{d+1}$
Required: Loss function $\ell(\beta)$, Embedding function ϕ
Output: Reconstructed Sample $\tilde{x}$
Estimate the Hessian $\hat{H}$ using Eq. (16) (for linear regression, this is $\frac{1}{m} X_{\text{pub}}^\top X_{\text{pub}}$)
if ℓ is MSE **then**
  Reconstruct the embedding $\tilde{z}$ using Eq. (7)
else
  Reconstruct the embedding $\tilde{z}$ using Eq. (15)
end if
if $\phi(x) = [x, 1]$ **then**
  Directly recover $\tilde{x} = \tilde{z}$
else
  Reconstruct the input $\tilde{x}$ using Eq. (8)
end if
Return first d dimensions of $\tilde{x}$

A. Algorithms

B. Additional Related Work

There is a large body of work on membership inference and reconstruction attacks on static models that is too long to survey here; see e.g. (Shokri et al., 2017; Carlini et al., 2021; 2022; Bertran et al., 2023; Carlini et al., 2023) for exemplars. This line of work focuses on attacking very large models, and the techniques developed within it fail when applied to simple models; see Appendix C. Our point of departure is that we focus on very simple models — but our attacker has access to two versions of the model, from both before and after a deletion operation. Several papers (Fredrikson et al., 2014; Wu et al., 2015) give "model inversion" attacks on linear models. These kinds of attacks are significantly weaker; given a model f, they infer partial information about x given the remainder of x and $f(x)$, rather than from the parameters of f.

Several papers study privacy attacks on pairs of models that result from an update. The most closely related is Chen et al. (2021), which was the first to explicitly note the added privacy risk inherent in machine unlearning, and give a membership inference attack that can be launched on pairs of models that result from an unlearning operation. Their membership inference attack is based on constructing shadow models, and they evaluate it on a range of model types, with performance increasing with model complexity. In comparison to Chen et al. (2021), our goal is reconstruction, which aims to recover the deleted point entirely, rather than membership inference, which aims only to recover a single bit of information (the presence or absence of an attack point in the training data). Salem et al. (2020) give reconstruction attacks in the context of single-gradient model updates that attempt to recover the point used to update the model. Their adversary operates in a more constrained setting than ours does, and does not have access to the model parameters—only API access to the model, and they attack a much weaker form of model update. They give an attack based on training shadow models and an auto-encoder to reconstruct samples used to update complex models (convolutional neural networks). In comparison, we assume that the adversary has access to the model parameters, but our attack can be used to reconstruct data points from much simpler models even when they are fully retrained; our attacks are also much more computationally efficient.

Finally, we mention Balle et al. (2022) which gives reconstruction attacks in a different model in which the adversary has more information than the standard attack scenario — in particular, the adversary studied in Balle et al. (2022) knows *the*

entire training set except one example and aims to reconstruct the unknown example.

C. Membership Inference Attacks on Linear Models

Linear models usually have lower privacy risks compared to neural networks because the parameters of a linear model are significantly fewer. To demonstrate the low privacy risks, we conduct a state-of-the-art membership inference attack (MIA) proposed by (Carlini et al., 2022) — Likelihood Ratio Attack (LiRA) — on the same tabular tasks. The goal of MIA is to determine whether a sample is in the training set of the target model.

On each task, we split the dataset into three splits, including 40% for training the target model, another 40% as the public samples for learning shadow models, and the rest 20% as the holdout set for evaluation. After training the target model on the private training set, we train 64 shadow models using the same optimization algorithm on the public samples with Bootstrap. Then, we use the joint set of the private samples and the holdout samples as samples under attack, and evaluate the attack performance.

As shown in Figure 2, the attack performance is close to random guessing, which implies that it is already challenging to determine which sample has been used in training when the target model is linear.

Figure 2: Membership inference attacks on ACS tasks.

D. Experiments

We evaluate our attack on both tabular and image data and on both classification and regression tasks. To simulate the process of deleting a single sample from a trained model, we first train a model with tuned hyperparameters on $X_{\text{priv}}, y_{\text{priv}}$ to obtain β^+, and then retrain the model with the same hyperparameters but on $X_{\text{priv}} \backslash x$, $y_{\text{priv}} \backslash y$ to obtain β^-. We note here that we do not use any approximate "machine unlearning" methods to obtain β^- and rather directly implement the "full retraining" gold standard.

Our attack requires public samples from the training data distribution but does not require any knowledge of the deleted data point (either its features or its label). We compare our attacks to two baselines that also make use of public data:

"**Avg**": $\hat{x} = \frac{1}{m} \sum_{x \in X_{\text{pub}}} x$. This "sanity check" baseline simply computes the average of the public samples as its guess for the deleted sample.

"**MaxDiff**": $\hat{x} = \arg\max_{x \in X_{\text{pub}}} ||x^T(\beta^+ - \beta^-)||$. This attack finds the sample among all public samples that leads to the maximum prediction difference across the two models, and uses this as its guess for the deleted sample.

These baseline attacks exploit the potential similarity between public and private training data. The "MaxDiff" baseline additionally incorporates information from the parameter change between the models. Since Salem et al. (2020) considers a different threat model to ours (black box access) and a weaker update model (single gradient step over updated sample), we show a comparison in Appendix E

Each dataset is randomly split into two halves, one of which serves as the private training data, and the other the public

samples. For each sample within the private training data, we retrain a model without that sample, and launch our attack (as well as the baseline attacks) in an attempt to recover that sample.

To emulate realistic scenarios for the model maintainer, when estimating β^+, we optimize the hyperparameter — the strength of l2 regularization λ — on the private training set $\{X_{\text{priv}}, y_{\text{priv}}\}$, and keep it fixed when estimating β^-. Additional results on attacking unregularized models are presented in Appendix F. The raw classification performance of the target models is shown in Appendix G.

We evaluate the success of individual attacks using cosine similarity between the deleted and reconstructed samples. We plot the Cumulative Distribution Function (CDF) of cosine similarity values between deleted samples and reconstructions, across all samples in the private data; a sharp increase near 1 implies near-perfect reconstruction.

D.1. Image Data

We use three simple image classification datasets, Fashion MNIST (FMNIST) (Xiao et al., 2017), MNIST (LeCun et al., 1998), and CIFAR10 (Krizhevsky et al., 2009). FMNIST and MNIST consist of 28×28 grayscale images, while CIFAR10 contains $32 \times 32 \times 3$ RGB images; the objective in all cases is 10-way classification. As preprocessing, we normalized the input features to the $[-1, 1]$ range. We attack several kinds of models:

Cross-entropy Loss for Multiclass Classification We attack a linear model with a softmax nonlinearity, trained to minimize the standard cross-entropy loss $\ell_{CE}(\beta, x, y) = -\log \sigma_y(x^\top \beta) + \lambda ||\beta||_2^2$.

We attack this model using Algorithm 2; in this setting, we do not need to run Eq (8) since the model contains a linear layer over the raw inputs x, and, therefore, the estimated gradient of the deleted sample w.r.t. the parameters β^+ is proportional to the deleted sample as described in Eq. (17). The Hessian matrix is empirically estimated from X_{pub}.

Ridge Regression with Random Features We attack a linear model over an embedding function ϕ, which generates random Fourier features (Rahimi & Recht, 2007), as $\ell_{Ridge}(\beta, \phi(x), y) = ||\phi(x)^\top \beta - y||_2^2 + \lambda ||\beta||_2^2$. This formulation has an analytical solution for the Hessian w.r.t. $\phi(x)$, which is $H = \phi(X)^\top \phi(X)$, but requires embedding inversion using Eq. (8).

Cross-entropy Loss with Random Features This combines the difficulty of both of the above settings: a lack of access to a closed-form solution for the model update, with the need to invert an embedding.

Figure 3 Shows the performance of our attack for all three model types on Fashion MNIST, MNIST, and CIFAR10. We consistently recover samples that are highly similar to the deleted sample for all considered model types. Figures 1, 4 and 5 show randomly sampled deletions, alongside their recovered closest match according to HRec and MaxDiff on CIFAR10, Fashion MNIST, and MNIST respectively in the most challenging scenario (cross entropy loss over random Fourier features). We show the results for the other model types in Appendix H.

D.2. Tabular Data

We use an income prediction task defined over data from the American Community Survey (ACS) (Ding et al., 2021) (2018 data). ACS Income data includes numerical income values which we use as the target for a regression task; on the same data, we can define a classification task by binarizing the numerical income - whether the income is higher than 50K or not. We preprocess the data with one-hot encoding of categorical columns and normalization of numerical columns so that all values lie between 0 and 1.

Ridge Regression for Income Prediction. The attack here is straightforward, and we directly apply Eq. (4) and the intercept normalization trick. Plots in the first row of Figure 6 illustrate the attack performance. If we knew the covariance matrix of the private data and λ, our attack in this setting would be guaranteed to perfectly recover the deleted sample. Hence our only source of estimation error comes from the fact that we need to estimate the covariance matrix from public samples and act as if $\lambda = 0$. As our results show, we still obtain nearly perfect reconstruction.

Ridge Regression with Random Features. Rather than performing ridge regression on the raw features, we now attack a ridge regression model trained on top of random Fourier features (Rahimi & Recht, 2007). We assume the random Fourier features are known to the attacker, along with the model weights (both of these would be needed to be able to evaluate the model).

We first apply our attack to reconstruct the embedding $\tilde{z}$ of the deleted sample, and then we solve an inverse problem to

reconstruct the original features. The attack performance is visualized in the second row of Figure 6; our method significantly outperforms both baselines, obtaining almost perfect reconstruction.

Binary Classification for Income Level Prediction. For this binary classification task, we attack two kinds of models: logistic regression and support vector machines (SVMs).

Both logistic regression and SVMs with the squared hinge loss have analytical expressions for their Hessian matrices, and they have the following form:

$$\hat{H} = X_{\text{pub}}^{\top} D X_{\text{pub}}, \tag{19}$$

where $D \in \mathbb{R}^{M \times M}$ is a diagonal matrix. For logistic regression, the diagonal terms are $D_{ii} = \sigma(x_i^{\top}\beta^{+})(1 - \sigma(x_i^{\top}\beta^{+}))$, where x_i is i-th sample in the public data, and σ is the sigmoid function; for SVMs, it is $D_{ii} = \mathbb{1}(1 - y_i x_i^{\top}\beta^{+} \geq 0)$, where $\mathbb{1}$ is the indicator function. Thus, the reconstruction can be obtained by computing $\hat{H}(\beta^{+} - \beta^{-})$ and using the intercept normalization trick.

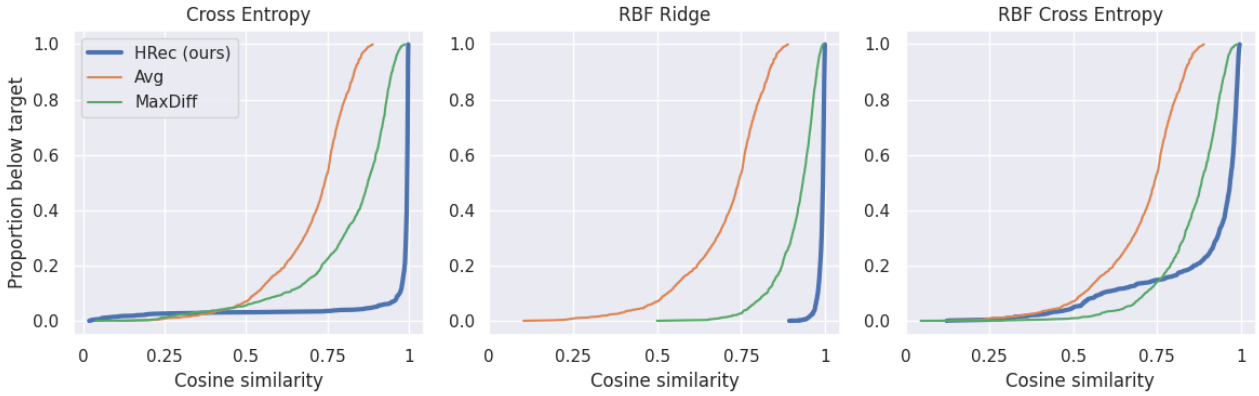
Figure 7 shows the performance of these attacks. Our attack HRec outperforms the baselines across all datasets and model classes, showing that our attacks remain successful when launched against simple tabular classification models.

Binary Classification with Random Features. We also attack binary classification models trained over an enriched set of random Fourier features. Figure 8 presents performance curves for our attack, as well as baselines. Once again we see that our attack performs exceptionally well. When attacking logistic regression models, our attack outperforms all baselines. When attacking SVM models the MaxDiff baseline also performs quite well.

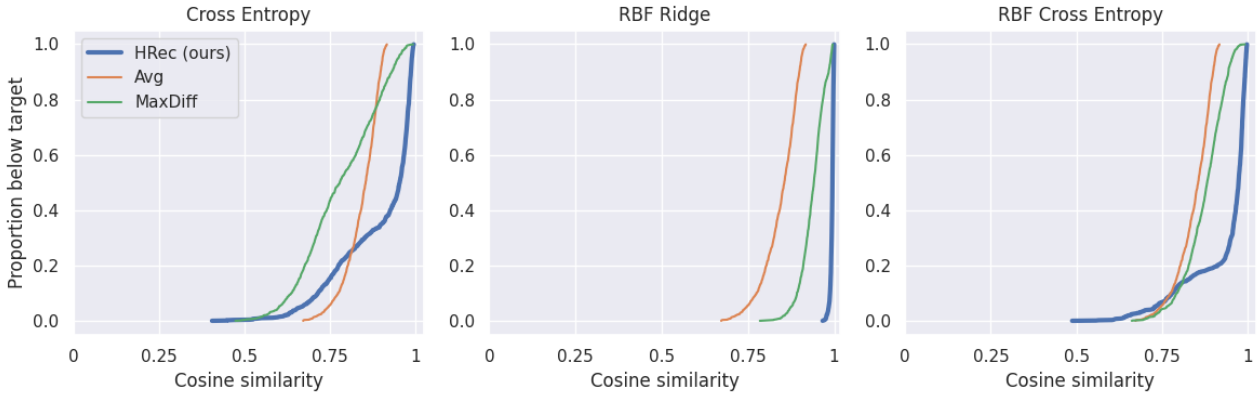
E. Additional Comparisons

Here we provide a limited comparison of Updates-leak (Salem et al., 2020) against our method and baselines for the same simple model architecture (cross-entropy loss over a linear model on top of 4096 random Fourier features). We stress that the threat model for Updates-Leak differs from our own in two important ways. First, they assume query access to the model, while we assume access to the parameters. Second, we carry out our attack on two models, fully trained to convergence on two different datasets, ($(X_{\text{priv}}, y_{\text{priv}})$ and $(X_{\text{priv}} \setminus x, y_{\text{priv}} \setminus y)$). In contrast, Updates-leak instead attacks the difference between a model trained on $(X_{\text{priv}} \setminus x, y_{\text{priv}} \setminus y)$ and an updated model where in the update, only a single gradient descent step is taken on the 'update' sample (x, y) (single sample attack version). The Updates-Leak approach also incurs a significantly higher computational cost due to its shadow model and encoder learning approach. For these reasons, we limit the comparison to a single model architecture on CIFAR10 while stressing this comparison is not 'apples to apples'. In particular, even though we plot the reconstrution cosine similarity curves on the same axis (and see that ours improves), our technique and UpdatesLeak are attacking *different pairs of models* (we attack the model that results from full retraining, whereas they attack the model that results from a single gradient update).

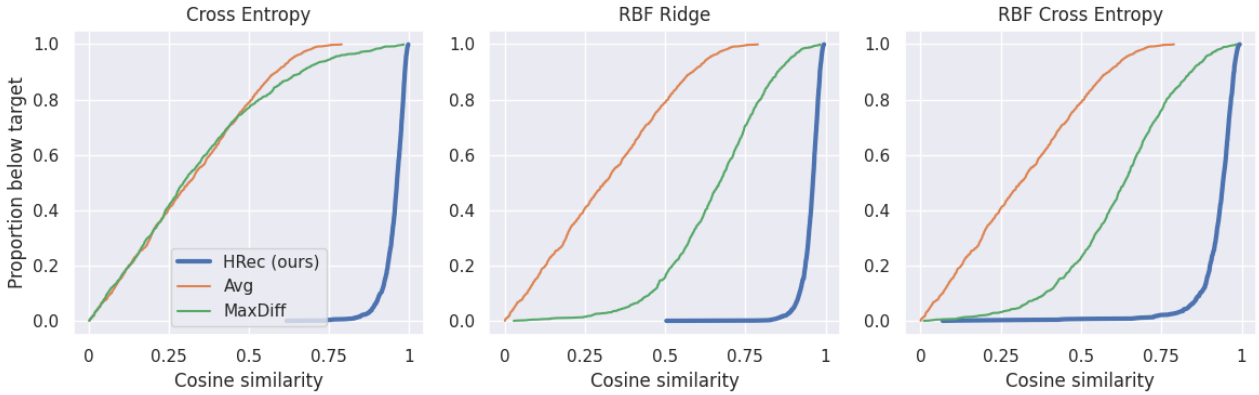
Figure 9 shows the cosine similarity comparison, and Figure 10 show some example reconstructions for Updates-Leak in this scenario. We leveraged their publicly available code to produce these comparisons, using their default configuration ($10,000$ shadow models are used for training, and their DC-GAN generator is trained for $10,000$ epochs on the shadow model dataset).



(a) Fashion MNIST results



(b) MNIST results



(c) CIFAR10 results

Figure 3: Cumulative distribution function of cosine similarity between the target (deleted) sample and the reconstructed sample via the average, MaxDiff, and HRec (our) attack on Fashion MNIST, MNIST, and CIFAR10 for three model architectures (linear cross-entropy, ridge regression over 4096 random Fourier features, and cross-entropy over 4096 random Fourier features). Here lower curves dominate higher curves. Our attack achieves better cosine similarity with the deleted sample across all settings; the effect is especially apparent in the denser CIFAR10 dataset.

Figure 4: Sample reconstructions on Fashion MNIST for a $40K$ parameter model (cross-entropy over random Fourier features of the raw input). We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. HRec produces reconstructions that are highly similar to the deleted images.

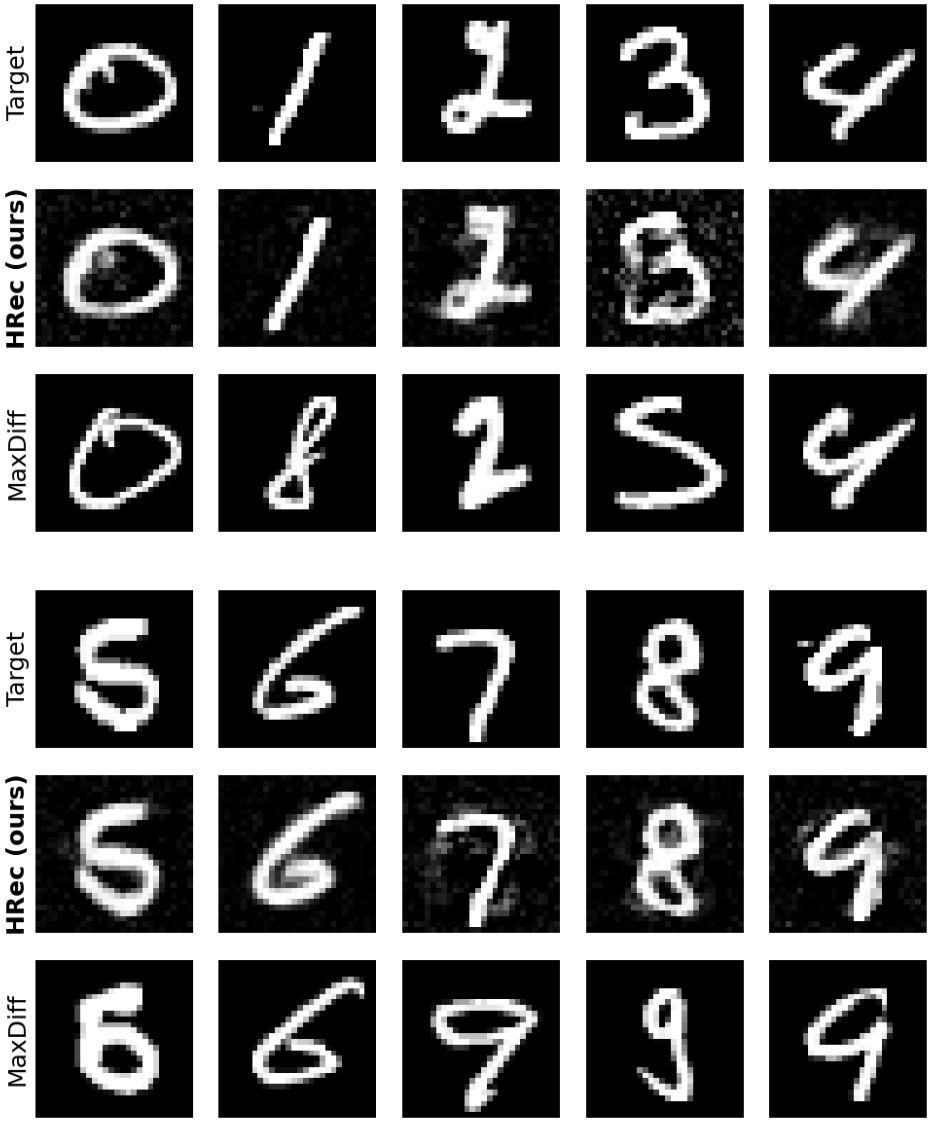


Figure 5: Sample reconstructions on MNIST for a $40K$ parameter model (cross-entropy over random Fourier features of the raw input). We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. Reconstructions from HRec are more similar to the deleted images than those from MaxDiff.

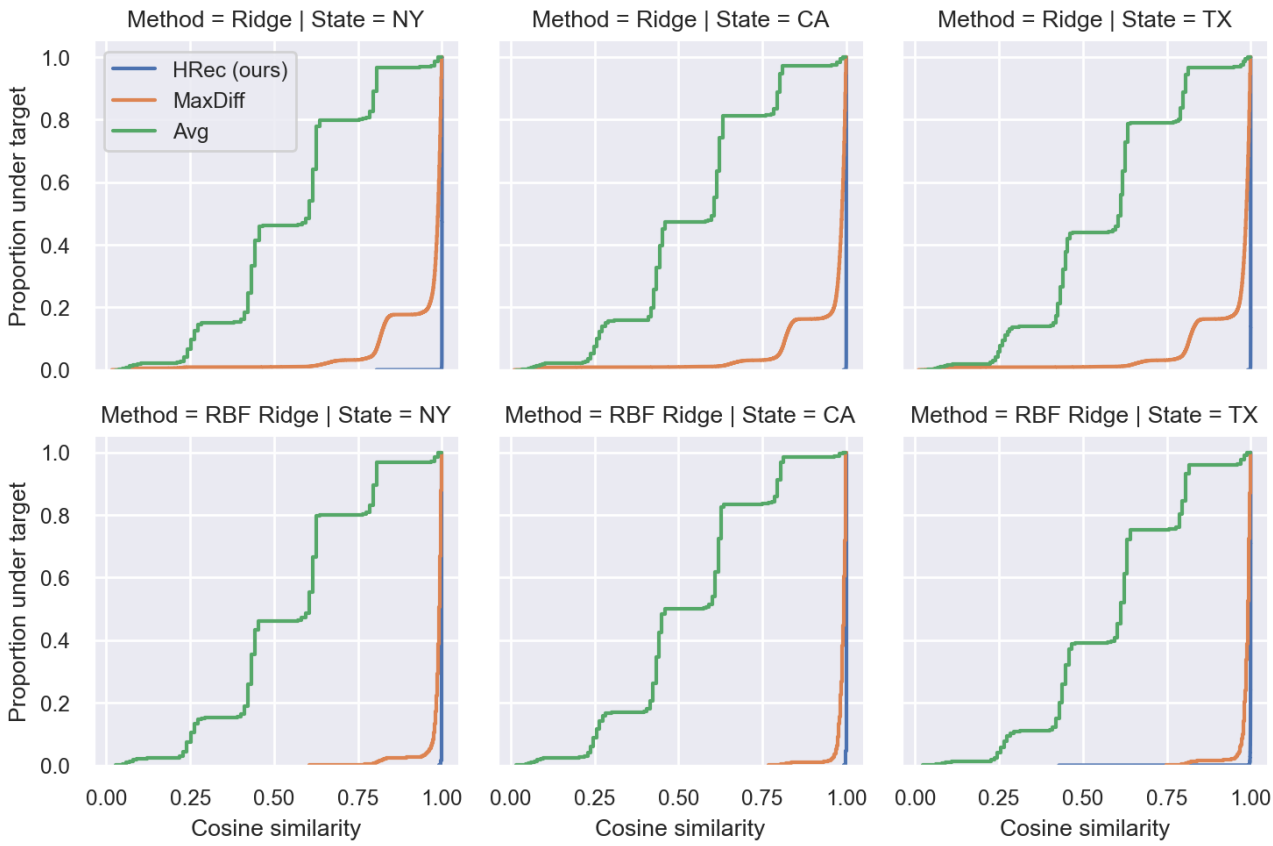


Figure 6: ACS Income Regression. Target models are ridge regression with tuned hyperparameters on original features (first row), and over random Fourier features (second row). ACS Income data from three states are used to demonstrate the effectiveness of our attack. Given the analytical single-sample update rules of linear regression, our attack (HRec) reconstructs the deleted sample almost perfectly on all datasets and different embedding functions.

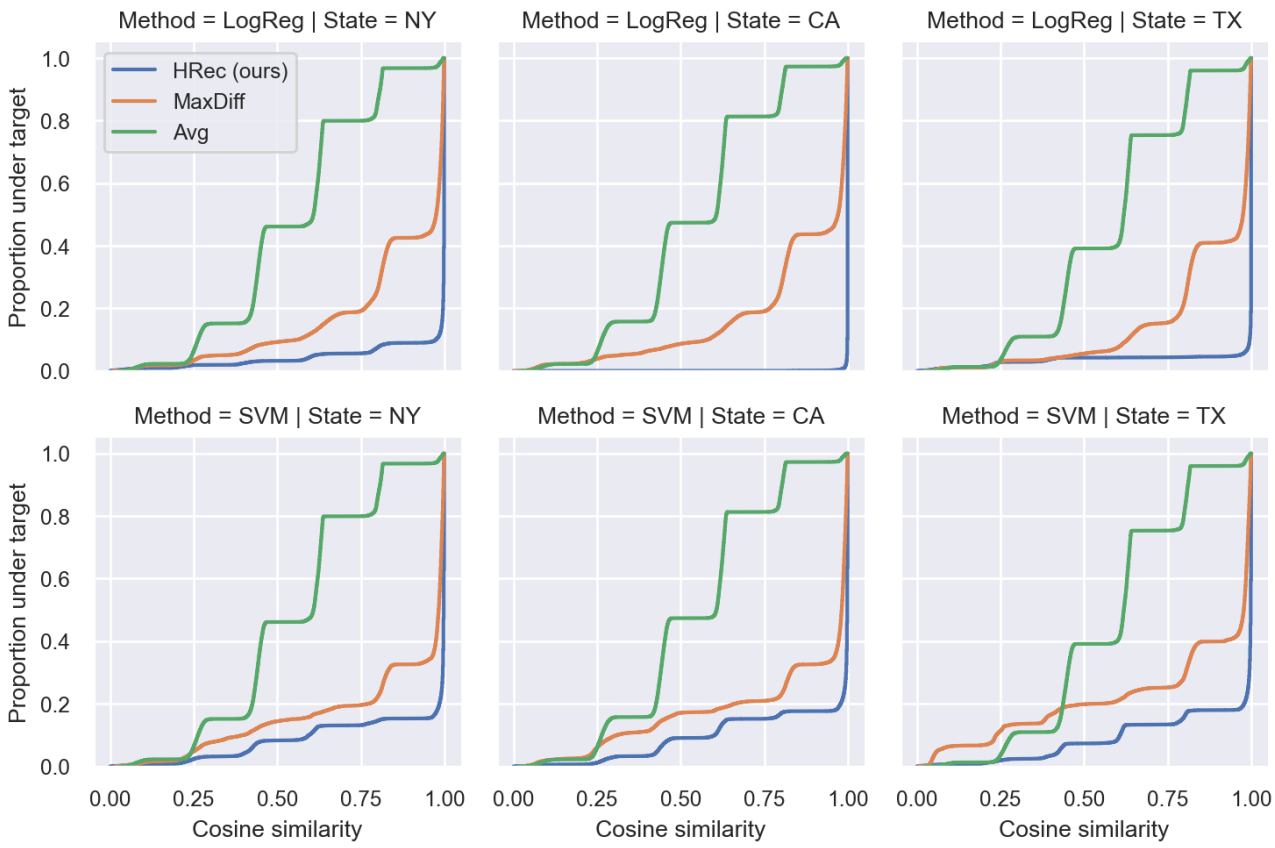


Figure 7: ACS Income Level Prediction. Target models are logistic regression (first row) and SVM (second row) for binary classification tasks. The analytical form of the single-sample update doesn't exist anymore, however, our approximation using Newton's update facilitates the outstanding performance of HRec among all attacks. Even though the reconstruction is not perfect due to approximation errors, a large number of deleted samples can still be reconstructed with high similarity scores.

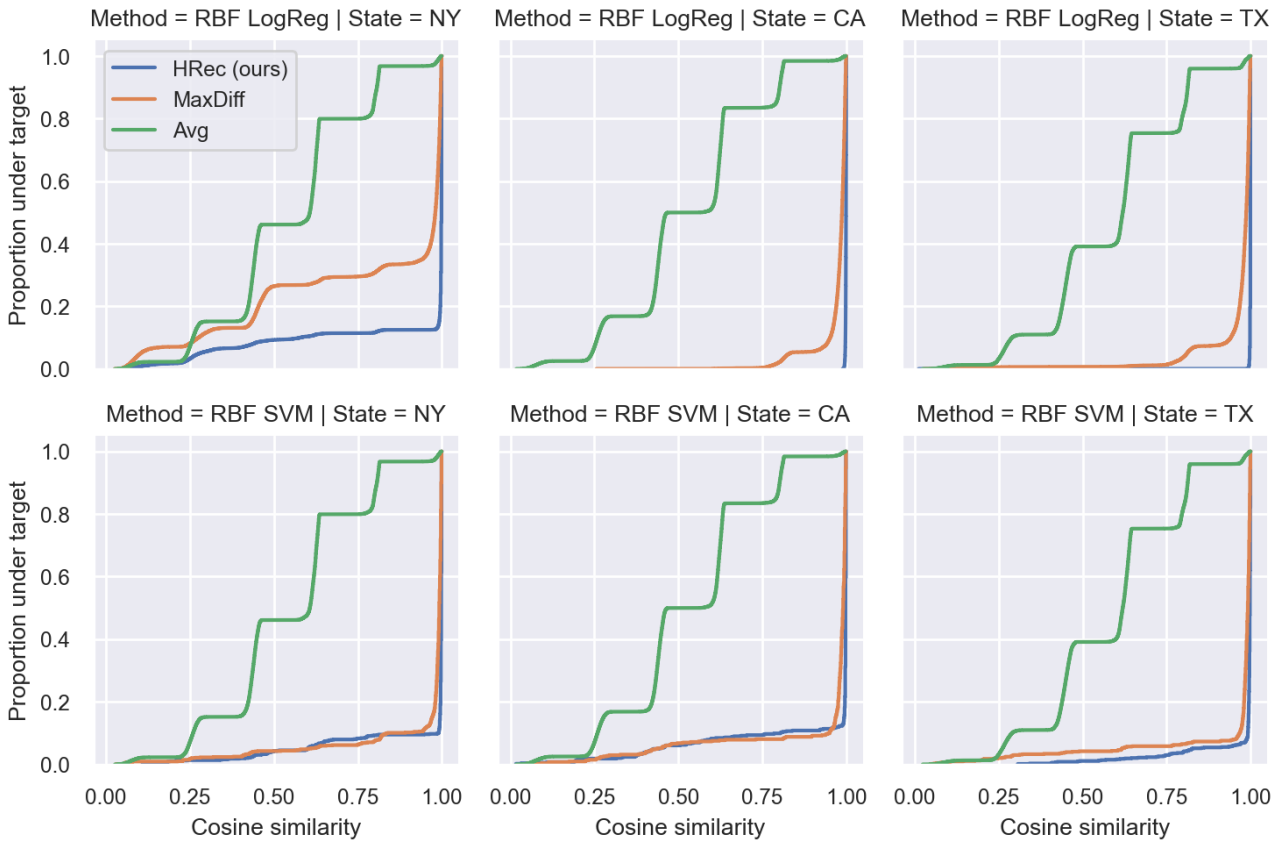
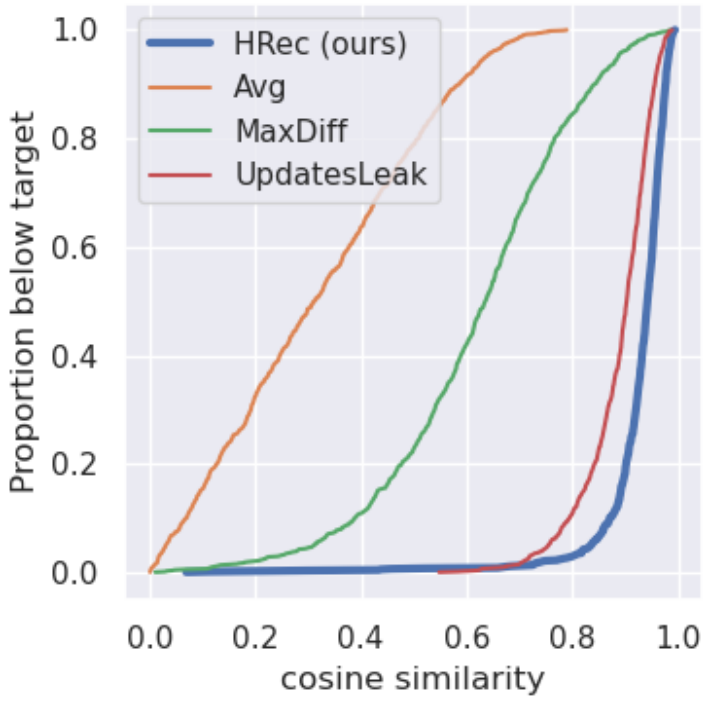


Figure 8: ACS Income Level Prediction using Random Fourier Features. Target models now are augmented with fixed random features. HRec still outperforms the two comparison partners when the target model is Logistic Regression, and performs similarly to MaxDiff when SVM is used.



(a) CIFAR10

Figure 9: Cumulative distribution function of cosine similarity between the target (deleted) sample and the reconstructed sample via the average, MaxDiff, Updates-Leak, and HRec (our) attack on CIFAR10 on a simple model (cross-entropy loss over a linear model on top of 4096 random Fourier features). **All attacks save for Updates-Leak operate against the full retraining baseline, that is the comparison between two models trained from scratch until convergence in a dataset with and without the 'deleted' sample ($(X_{\text{priv}}, y_{\text{priv}})$ and $(X_{\text{priv}} \setminus x, y_{\text{priv}} \setminus y)$), Updates-Leak instead attacks two models, one trained until convergence on the dataset without the sample ($(X_{\text{priv}} \setminus x, y_{\text{priv}} \setminus y)$), and one that took a single gradient descent step on the loss of the updated sample x, y.** Here lower curves dominate higher curves. Our attack achieves better cosine similarity with the deleted sample.

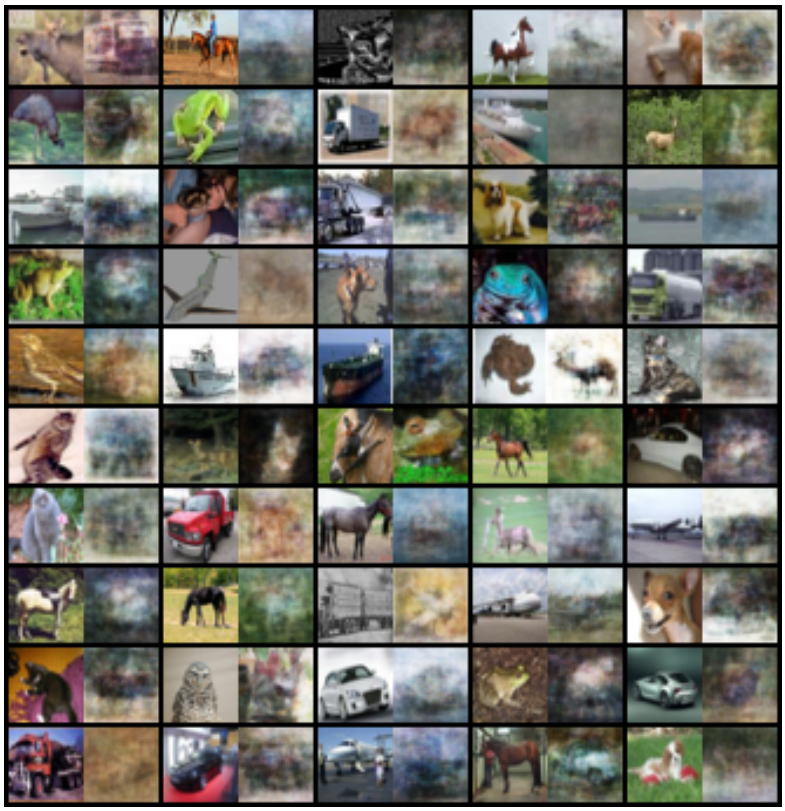


Figure 10: Sample reconstructions for Updates-Leak on CIFAR10. Original images and their corresponding reconstruction are shown side by side in an alternating fashion. The model architecture is cross-entropy loss over a linear model on top of 4096 random Fourier features. The models before and after the update differ in a single gradient descent step being taken on the update sample (x, y).

F. Attacking Unregularized Models

In our main experiments, we emulate the more realistic situation where the model maintainer tunes the hyperparameter of the target model on the entire dataset, and keeps it fixed during unlearning. Since the impact of regularization on the attack performance is rather challenging to analyze and not immediately obvious, we here present results on attacking models without regularization.

F.1. ACS Income Regression

On this task, the model maintainer directly optimizes the objective Equation (1) without the regularization term:

$$\beta^* = \arg\min_{\beta} \|X\beta - y\|_2^2 \tag{20}$$

This problem admits an analytical expression for β^+ and β^-, which can be written as:

$$\beta^+ = C^{-1} X_{\text{priv}}^\top y_{\text{priv}}, \tag{21}$$

$$\beta^- = (C - xx^\top)^{-1}(X_{\text{priv}}^\top y_{\text{priv}} - x^\top y), \tag{22}$$

where $C = X_{\text{priv}}^\top X_{\text{priv}}$ is the covariance matrix. In the scenario where the inverse of the covariance matrix doesn't exist, we use the Moore–Penrose inverse instead. Our attack still stays the same.

The results are presented in Figure 11, and we can see that without regularization

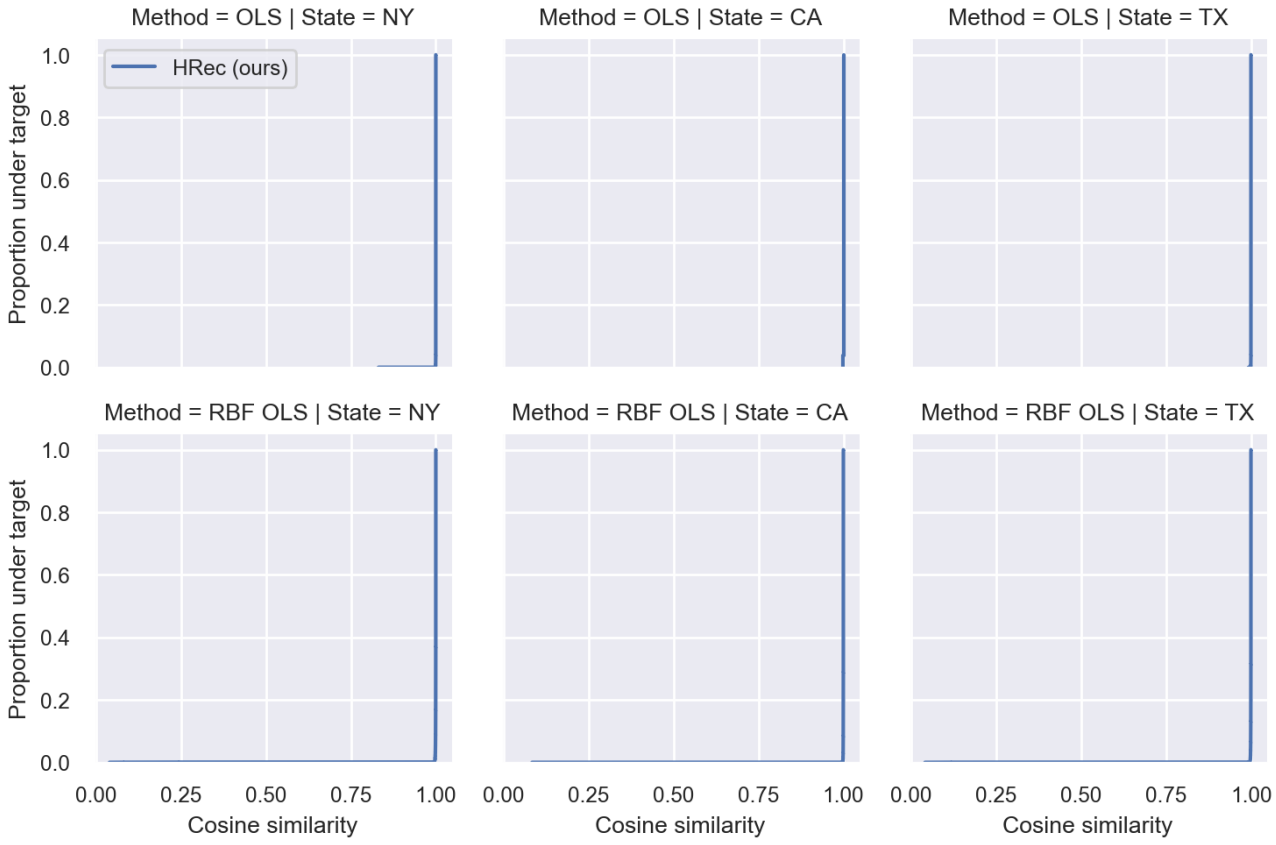
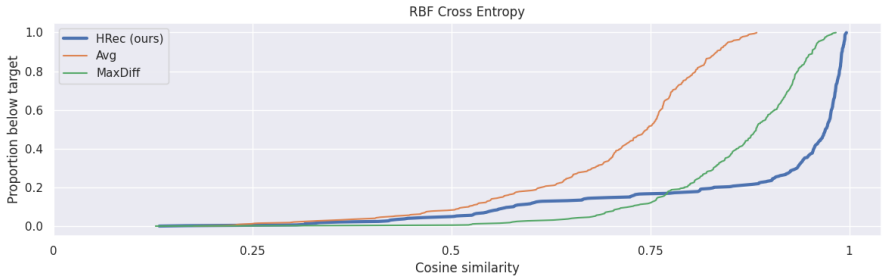


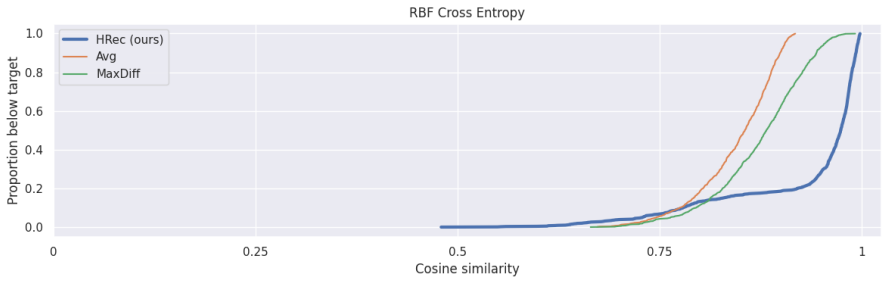
Figure 11: ACS Income Regression. Target models are ordinary linear regression on original features (first row), and over random Fourier features (second row). Our attack HRec reconstructs the deleted sample almost perfectly.

F.2. Image Classification Tasks

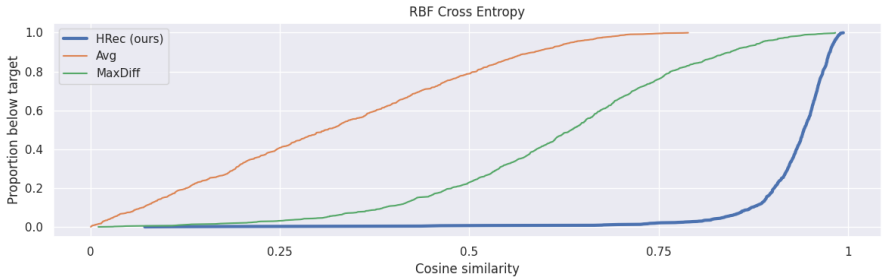
Here we additionally show results across CIFAR10, MNIST, and Fashion MNIST on the more challenging target model scenario (RBF Cross Entropy). For these results, the model maintainer does not use any form of regularization. Results are shown in Figure 12.



(a) Fashion MNIST results



(b) MNIST results



(c) CIFAR10 results

Figure 12: Cumulative distribution function of cosine similarity between the target (deleted) sample and the reconstructed sample via the average, MaxDiff, and HRec (our) attack on Fashion MNIST, MNIST, and CIFAR10 for a target model using cross-entropy over 4096 random Fourier features. In this scenario, the model maintainer does not use any form of regularization when training the original or updated model. Here lower curves dominate higher curves. Our attack achieves better cosine similarity with the deleted sample across all settings; the effect is especially apparent in the denser CIFAR10 dataset.

G. Performance of Target Models

G.1. ACS Income Tasks

Table 1: Performance of target models on ACS Income tasks. Regression tasks are evaluated using r^2, which indicates the portion of explained variance, and classification tasks are evaluated using $F1$ score since class labels are not balanced.

State		NY		CA		TX	
Task	Target model		+RBF		+RBF		+RBF
ACS Income Regression (r2)	Ridge Regression	0.2755	0.32712	0.30139	0.3510	0.3089	0.3510
ACS Income Level Classification (F1)	Logistic Regression	0.7154	0.7230	0.7313	0.7413	0.6889	0.7009
	Linear SVM	0.7099	0.7149	0.7345	0.7325	0.6868	0.6968

G.2. Image tasks

Table 2: Out of sample accuracy of target models on Image tasks

Dataset	Linear Cross-entropy	RBF Ridge	RBF Cross Entropy
CIFAR10	39.5%	48.7%	50.4%
MNIST	91.6%	96.2%	96.4%
Fashion MNIST	84.5%	87.3%	88.4%

H. Additional Results on Image Datasets

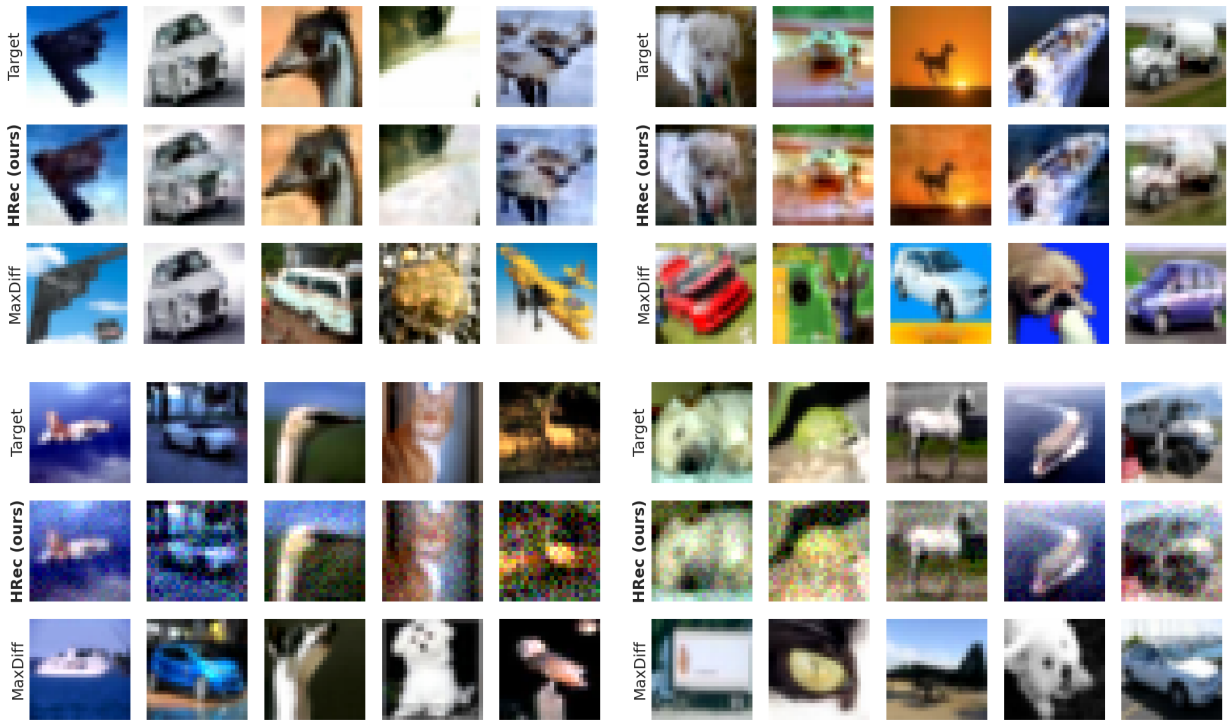


Figure 13: Sample reconstructions on CIFAR10. Rows 1-3 rows show results of attacking a linear cross-entropy model, and rows 4-6 show similar results for ridge regression over 4096 random Fourier features. We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. HRec produces reconstructions that are highly similar to the deleted images.

Figure 14: Sample reconstructions on Fashion MNIST. Rows 1-3 rows show results of attacking a linear cross-entropy model, and rows 4-6 show similar results for ridge regression over 4096 random Fourier features. We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. HRec produces reconstructions that are highly similar to the deleted images.

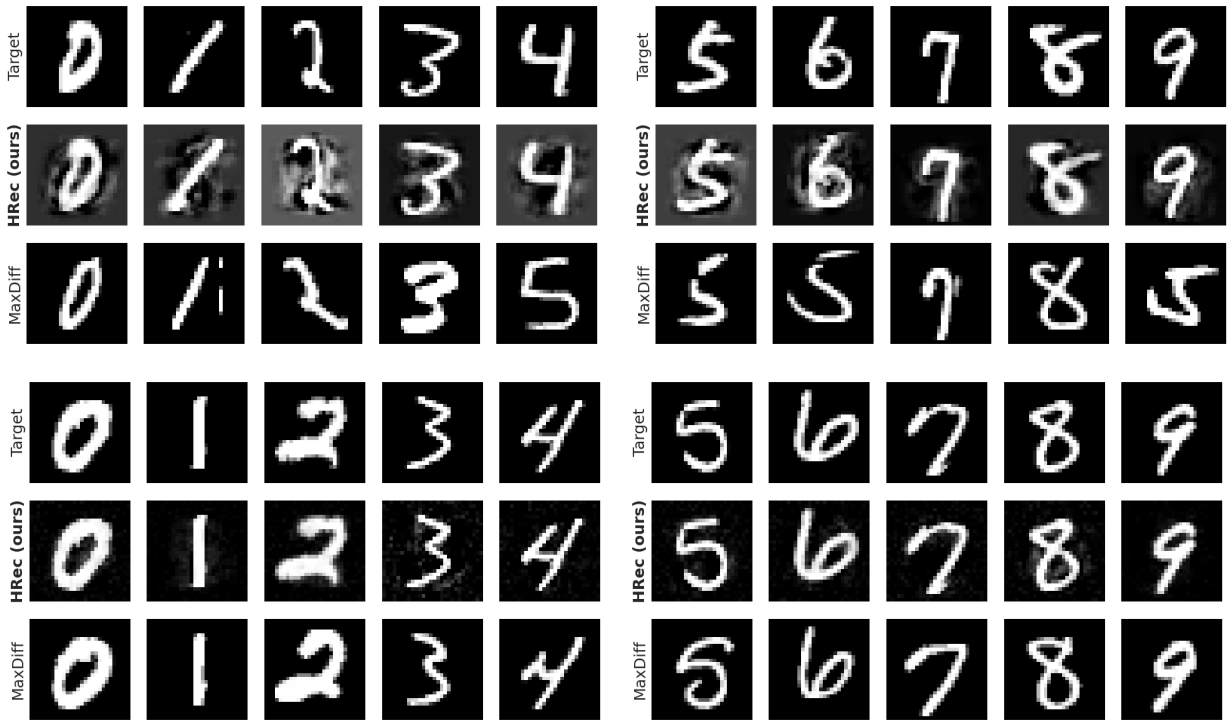


Figure 15: Sample reconstructions on MNIST. Rows 1-3 rows show results of attacking a linear cross-entropy model, and rows 4-6 show similar results for ridge regression over 4096 random Fourier features. We randomly chose one deleted sample per label (shown in rows 1 and 4) and compared them against the reconstructed sample using our method (HRec, rows 2 and 5) and a perturbation baseline (MaxDiff, rows 3 and 6) which searches for the public sample with the largest prediction difference before and after sample deletion. HRec produces reconstructions that are highly similar to the deleted images.