Finding Solutions to Generative Adversarial Privacy

Dae Hyun Kim[*]
dhkim16@stanford.edu

Taeyoung Kong[†]
kongty@stanford.edu

Seungbin Jeong[†]
sjeong91@stanford.edu

Abstract

We present heuristics for solving the maximin problem induced by the generative adversarial privacy setting for linear and convolutional neural network (CNN) adversaries. In the linear adversary setting, we present a greedy algorithm for approximating the optimal solution for the privatizer, which performs better as the number of instances increases. We also provide an analysis of the algorithm to show that it not only removes the features most correlated with the private label first, but also preserves the prediction accuracy of public labels that are sufficiently independent of the features that are relevant to the private label. In the CNN adversary setting, we present a method of hiding selected information from the adversary while preserving the others through alternately optimizing the goals of the privatizer and the adversary using neural network backpropagation. We experimentally show that our method succeeds on a fixed adversary.

1 Introduction

Assume that an entity, whom we call the *privatizer* holds some data it wishes to publish to the public, but that the data includes some sensitive information, such as gender or race, that the entity wants to hide from the public. We will assume that there exists another entity, whom we call the *adversary*, whose goal is to recover the sensitive information from the data published by the privatizer, and publicly available data that relates the features of the privatizer's data with the sensitive labels.

If the privatizer naively removes the sensitive labels and publishes the rest of the data as-is, the adversary can accurately recover the sensitive labels using various machine learning models with the publicly available data. Thus, the privatizer needs to encrypt its data using a privatization function before it publishes it to the public. We will assume the worst case for the privatizer, in which the privatization function is known to the adversary, perhaps through publications or information leakage. The natural methodology that the adversary would use is to first simulate the privatization function on the publicly available data to obtain a relationship between the privatized data and the sensitive information. Then he would use this relationship to predict the sensitive

labels from the privatized data published by the privatizer. The privatizer's goal is to hinder such attempt by the adversary.

This problem is a zero-sum game, in which the adversary attempts to minimize his loss on predicting the sensitive labels, while the privatizer attempts to maximize this value. From the privatizer's perspective, finding the optimum requires solving a maximin problem:

$$\operatorname*{argmax}_{P} \min_{f \in \mathcal{H}_{adv}} \mathcal{L}(f(P(X)), y),$$

where P stands for the privatizer's function and L stands for the loss function. In addition, the privatizer wants to release useful data to the public, which we will model by restricting the amount of change in the data:

$$\|X - P(X)\|_2^2 \le D,$$

for some constant D. In this paper, we investigate this problem when $\mathcal{H}_{adv}$, the hypothesis class of the adversary, is either linear or convolutional neural network (CNN).

2 Related Work

2.1 Minimax Optimization

Zero-sum games are games in which the two players' utilities sum to zero, which is commonly seen in real life. The two players in a zero-sum game are subject to solving a minimax game, in which each of the players attempt to minimize the maximum utility gained by the opponent. Due to its omnipresence, much work has been done to accurately and efficiently solve the minimax games under variant restrictions [1, 2].

The setup that we are working with is a zero-sum game between the privatizer and the adversary, and this naturally gives rise to a maximin game, which is just a negation of a minimax game.

2.2 Generative Adversarial Networks

Recently, *generative adversarial network* (GAN) [3, 4] was proposed to generate data to simulate real data. The GAN solved the minimax problem between the *generator* and the *discriminator* to train a good generative network. The idea of training a generator with another network inspired us on how to train the privatizer network. This generator tries to deceive the adversary, while the adversary tries to avoid being fooled by it.

---

[*]Department of Computer Science, Stanford University
[†]Department of Electrical Engineering, Stanford University

2.3 Generative Adversarial Privacy

Recent works [5, 6] have introduced the idea of *generative adversarial privacy* (GAP) and showed that it is possible to add noise that selectively disturbs algorithms that can learn the *private labels*, while preserving the utility of algorithms that can learn *public labels*. There is also some promising results in the case in which the models used by the adversary and the ally are both linear [7].

Our work differs from these previous works in that we aim to preserve the overall data, instead of choosing a label to preserve the accuracy for.

3 Methods

3.1 Linear Adversary Case

We will solve the privatizer's problem with lossy compression using a compression matrix A. In this case, the privatizer's problem simplifies to:

$$\text{Minimize} \quad \left\| y^T \left(XAA^+ \right) \left(XAA^+ \right)^+ \right\|_2^2,$$
$$\text{subject to} \quad \left\| X - XAA^+ \right\|_{Frob}^2 \leq D$$

where the compression matrix A is the variable. This problem is difficult because the variable is encapsulated in a pseudoinverse that is difficult to simplify.[1]

We therefore look at a smaller subset of the problem by restricting the data compression with respect to the features. Then, the problem reduces to partitioning the set of features into two sets, R, the set of features that are removed by the compression, and S, the set of features that are preserved by the compression. The optimum of this subproblem can be found by solving

$$\text{Maximize} \quad \left\| y - X_{i \in R} X_{i \in R}^+ \right\|_2,$$
$$\text{subject to} \quad \left\| X_{i \in R} \right\|_2^2 \leq D$$

where the set of excluded features, R, is the variable.[2]

Solving this problem by brute force results in exponential time complexity. Therefore, we propose an approximation algorithm for computing the optimum based on the ideas of the greedy algorithm. In each step of the greedy algorithm (Algorithm 1), we select the element that maximizes the utility, $\|y - X_{i \in R \cup \{e\}} X_{i \in R \cup \{e\}}^+\|_2$, per unit cost, $\left\| X_{R \cup \{e\}} \right\|_2^2$.

3.2 CNN Adversary Case

In the CNN adversary case, we assume that the adversary's network comprises of convolutional layers, activation layers (ReLu), pooling layers, fully connected layers, and a loss layer (softmax cross entropy loss) as exemplified by Figure 1a.

Algorithm 1 Greedy Algorithm

1: **procedure** GREEDY-APPROX(D)
2:      $R \leftarrow \phi$
3:      **do**
4:          $e_{next} \leftarrow$ FIND-NEXT (R, D)
5:          $R \leftarrow R \cup \{e_{next}\}$
6:      **while** $e_{next} \neq$ null
7:      **return** R
8: **end procedure**
1: **procedure** FIND-NEXT(R, D)
2:      $(u/c)_{max} \leftarrow -\infty$
3:      $e_{next} \leftarrow$ null
4:      **for** $e \in \{1, \cdots, n\} \setminus R$ **do**
5:          $c \leftarrow \left\| X_{R \cup \{e\}} \right\|_2^2$
6:          **if** $c > D$ **then**
7:              **continue**
8:          **end if**
9:          $u \leftarrow \|y - X_{i \in R \cup \{e\}} X_{i \in R \cup \{e\}}^+\|_2$
10:     **if** $v/c > (u/c)_{max}$ **then**
11:         $(u/c)_{max} \leftarrow u/c$
12:         $e_{next} = e$
13:     **end if**
14:     **end for**
15:     **return** e_{next}
16: **end procedure**

The privatizer uses a neural network (Figure 1b) that consists of an (1) encoder, which not only reduces the features by convolutional layers, but also tries to capture the high level features that the adversary would capture, a (2) decoder, which uses deconvolutional layers to map the captured features into the pixel space. The resulting output of the privatizer's network is the noise, ΔX that he/she adds to the original image X, to obtain a privatized image, $X + \Delta X$. The restriction in the amount of change in the data, in this case, becomes:

$$\|\Delta X\|_2^2 \leq D.$$

We also added a constraint that we want to preserve the prediction accuracy as much as possible for a selected set of labels with a pre-trained model on the original data, which we call *protected labels*. [3]

In determining the model for the privatization function, the privatizer solves the maximin problem mentioned in Section 1. The privatizer's method for solving this maximin problem by (Algorithm 2): (1) initializing the privatizer's internal model for the adversary's parameters, θ_{adv}, and the privatizer's own parameters, θ_{priv}, randomly, (2) fixing θ_{priv} and solves for the optimal θ_{adv} against the loss function of the private labels from the adversary's perspective using backpropagation, (3) fixing θ_{adv} and solves for the optimal θ_{priv} against the

---

[1] One of the steps leading to the convexity results in the milestone had an error.

[2] If $y = X\theta$, the target function reduces further to $\left\| \left(X_{i \in R} - X_{i \in S} X_{i \in S}^+ X_{i \in R} \right) \theta_{i \in R} \right\|_2^2$.

[3] We are able to preserve prediction accuracies of labels that are not highly correlated with the private label without this constraint, but with this constraint, we can preserve prediction accuracies on even highly correlated labels such as gender and decoration in 11k Hands.

(a) Adversary neural network structure

(b) Privatizer neural network structure

Figure 1: Adversary and privatizer network structures

Algorithm 2 Maximin Algorithm for CNN Adversary

1: **procedure** SOLVE-MAXIMIN(X, D)
2: $\quad\theta_{pro} \leftarrow \underset{\theta}{\operatorname{argmin}} \mathcal{L}_{pro}(X; \theta)$
3: $\quad\theta_{adv} \sim \text{Xavier}$
4: $\quad\theta_{priv} \sim \text{Xavier}$
5: $\quad$**while** $\Delta\theta_{adv} > \epsilon$ or $\Delta\theta_{priv} > \epsilon$ **do**
6: $\quad\quad\theta_{adv} \leftarrow \underset{\theta}{\operatorname{argmin}} \mathcal{L}_{priv}(X + \Delta X(X, \theta_{priv}, D); \theta)$
7: $\quad\quad\theta_{priv} \leftarrow \underset{\theta}{\operatorname{argmax}} \mathcal{L}_{priv}(X + \Delta X(X, \theta, D); \theta_{adv})$
8: $\quad\quad\theta_{priv} \leftarrow \underset{\theta}{\operatorname{argmin}} \mathcal{L}_{pro}(X + \Delta X(X, \theta, D); \theta_{pro})$
9: $\quad$**end while**
10: **end procedure**

---

dataset	instances (m)	features (n)	labels (N)
$\mathcal{H}_{adv}$: linear			
Beijing PM 2.5 [8]	41757	7	2
UCI Wine Quality [9]	4898	7	5
$\mathcal{H}_{adv}$: CNN			
GENKI [10]	4000	$200 \times 200 \times 1$	4
11k Hands [11]	5538	$200 \times 200 \times 3$	4

Figure 2: Summary of datasets

loss function of the private labels from the privatizer's perspective using backpropagation, (4) finding the optimal θ_{priv} against the loss function of the protected labels, and (5) repeating these steps until convergence.

4 Dataset and Features

For the linear adversary case, we use generated data of various sizes to confirm that the greedy approximation indeed approaches the true optimum. We generated the input data randomly using a uniform distribution unif $(0, 1)$, and varied the number of features from 4 to 7 and the number of instances from 10 to 1000.

We use Beijing PM 2.5 dataset [8] and UCI Wine Quality dataset [9] (Figure 2) to observe the properties of the optimum that we obtain. Of the properties included in the Beijing PM 2.5 dataset, we utilize the properties *year*, *month*, *day*, *hour*, *temperature*, *pressure*, and *cumulated wind speed* as features, and *PM2.5 concentration* and *dew point* as labels. For this dataset, we removed any instance that included missing features. In addition, of the properties included in the UCI Wine Quality dataset, we utilize the properties *fixed acidity*, *volatile acidity*, *citric acid*, *residual sugar*, *chlorides*, *free sulfur dioxide*, and *total sulfur dioxide* as features, and *density*, *pH*, *sulphates*, *alcohol*, and *quality* as labels.

In the investigation of CNN adversaries, we used two different datasets called GENKI [10] and 11k Hands [11] (Figure 2). GENKI is a dataset of 4000 human faces that is labeled with *head pose* and *smile content*. We replaced images that included multiple people or that were too low resolution for even humans to understand the image, and labeled *gender* for each of the images. We also normalized the sizes to 200×200 and converted the images to grayscale.

The main dataset is 11k Hands, a dataset of 11076 hands labeled with *the subject ID*, *gender*, *age*, *skin color*, *left/right*, which we will call *hand side*, *dorsal/palmar*, *accessories*, *nail polish*, and *irregularities*. We only used the dorsal side (5538) images and combined the labels accessories and nail polish by an 'or' operation to obtain a new label, *decoration*. Then, we reduced the size of the images to 200×200. We also augmented the training data by flipping the images, changing to grayscale, adding noise, and shifting the images to help the network capture essential features.

5 Results and Discussion

5.1 Linear Adversary Case

Figure 4 shows the relationship between the number of instances and the proportion of runs in which the greedy approximation achieved the optimum for varying number of features, obtained from the randomly generated toy data. The results indicate that the greedy approximation achieves the optimum more often as the number of instances increases. Furthermore, the results seem to indicate that for greater number of features, we need more instances in order to accomplish the same proportion.

In the experiment with the Beijing PM 2.5 dataset, we set the dew point as the private label and PM 2.5 as the public label. Figure 3a shows the change in the L2 loss for these two labels. The loss of the dew point increased from 0.044 to 0.153, while the loss of PM 2.5 increased only from 0.031 to 0.033. The features removed by greedy algorithm were in the order of: temperature, pressure, and day (of month). These features are rele-

(a) dew point (private) vs PM 2.5

(b) alcohol % (private) vs wine quality

(c) alcohol % (private) vs pH

(d) gender (private) vs smile content

(e) accuracies of labels for 11k Hands

Figure 3: Effect of privatization: (a) Beijing PM 2.5, (b, c) Wine (d) GENKI, and (e) 11k Hands

Figure 4: Ratio of optimum achieved by greedy

vant to both labels, but whereas the dew point is directly related to the temperature and the pressure, PM 2.5 is only indirectly related to these features.

We set alcohol content as the private label and the rest of the labels as public labels in the experiment with the UCI Wine Quality. Figure 3b shows the change in the L2 loss for the private label, alcohol content, and a public label, wine quality. The loss of the alcohol content increased from 0.096 to 0.108, while the loss of the wine quality increased only from 0.081 to 0.082. The features removed by the greedy algorithm were in the order of: total SO$_2$, fixed acidity, and citric acid. These features are highly relevant to the alcohol content.[4]

---

[4]SO$_2$ is a byproduct of fermentation, the cooler the origin of the wine, the higher the acidity and the lower the alcohol content.

On the other hand, if we compare the loss of the private label, alcohol content, against that of another public label, pH, we experience a relatively high increase in the loss of the public label (Figure 3c). Specifically, the loss of the pH increased from 0.055 to 0.068, which is comparable to the increase in the loss of the alcohol content. This is because two of the removed features, fixed acidity and citric acid, are highly correlated with both the pH and the alcohol content. Here, we conclude that it is more difficult to preserve accuracy of the public labels that are dependent on the same features as the private label.

5.2 CNN Adversary Case

We used Google TensorFlow [12] for handling neural networks.

For the GENKI dataset, we used two convolutional layers of sixteen 7×7 filters, each followed by a max-pool layers, and a fully connected layer of size 128 for the adversary's network. We used the same structure as the adversary's network for the privatizer's encoder, and used a fully connected layer of size 40000 for the privatizer's decoder.

We set gender as the private label, and the smile content as the public label in this dataset. The original test accuracy was 73% for the gender and 70% for the smile content. With the privatized images, the accuracy of the gender decreased to 53%, while that of the smile content increased to 77% (Figure 3d).

For the 11k Hands dataset, we used three convolutional layers with two 3×3 filters, four 4×4 filters and eight 3×3 filters respectively, each followed by a max-pool layer, and finally a fully connected layer of size 32 for the adversary's network. The pre-trained neural net-

Figure 5: Original and privatized images

work for the protected labels has three convolutional layers with four 3×3 filters, eight 4×4 filters and sixteen 3×3 filters respectively, each followed by a max-pool layer, and finally a fully connected layer of size 64. The encoder of the privatizer has the same structure as that of the protected labels, and decoder of the privatizer is symmetric to the encoder (Figure 1b).

We set gender as the private label, decoration as the protected label, and the rest as public labels. The accuracy achieved with the original data is 91.6% (gender), 89.5% (skin), 97.1% (hand side), and 97.2% (decoration). With the privatized images, the accuracy for gender dropped to 58.7%, while the accuracies for skin, hand side, and decoration changed to 89.5%, 97.0%, and 97.3%, respectively (Figure 3e). In sum, the accuracy for the private label dropped, whereas the accuracies for the other labels did not change significantly. As we desired, the privatized images did not undergo notable distortion (Figure 5).

Theoretically the convergence occurs when the privatizer can cover all the private label-related features to a certain level. However, due to the limited computing resources and limited time, we were not able to attempt a thorough set of parameters with larger networks, and did not observe the convergence.

6 Conclusion

We present an efficient method of constructing a privatizer when the adversary is limited to linear models using a greedy approximation, which improves with the number of instances increasing. Moreover, we found that the greedy algorithm removes features most correlated with the private label first, and that the algorithm preserves the predictability of public labels as long as they are sufficiently independent of the features that are relevant to the private label.

We also present a method for building a privatizer against a CNN adversary. We were able to selectively lower the private label's accuracy while preserving other labels' accuracies against a fixed CNN adversary. However, we lacked computational power for testing sufficiently many hyperparameters to achieve convergence.

7 Future Work

In the linear adversary case, theoretically bounding the performance of the greedy algorithm and finding how the relationship between features affects the optimum would solidify our findings. Also, we only compressed the data with respect to the standard basis, but because any orthonormal basis can be transformed into a standard basis using an invertible matrix multiplication, we can easily generalize our results to any other orthonormal basis. The natural next step is to find which orthonormal basis induces the best optimum.

Regarding the CNN adversary case, we will continue aiming for convergence of the algorithm by searching for hyperparameters in a larger space with greater computational power. In this case, the loss function should be calculated for all the history of the evolving adversary so that all the private attribute-related features are covered. Afterwards, whether our method generalizes to more complicated neural networks remains a future work.

Contributions

Dae Hyun Kim has taken the lead on the linear adversary case and helped out with the CNN adversary case through general advice and help with finding parameters. Taeyoung Kong has helped out with the linear adversary case through confirmation and scribing, and helped out with the CNN adversary case through general advice and help with finding parameters. Seungbin Jeong has taken the lead on the CNN adversary case and helped out with the linear adversary case through general advice.

References

[1] C. Charalambous and J. W. Bandler, "Nonlinear programming using minimax techniques," *Journal of Optimization Theory and Applications Volume 13*, 1974.

[2] C. Charalambous and A. Conn, "An efficient method to solve the minimax problem directly," *SIAM Journal on Numerical Analysis*, 1978.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[5] C. Huang, P. Kairouz, X. Chen, L. Sankar, and R. Rajagopal, "Context-aware generative adversarial privacy," *Entropy*, Submitted.

[6] J. Hamm, "Minimax filter: Learning to preserve privacy from inference attacks," *arXiv preprint arXiv:1610.03577*, 2016.

[7] K. Xu, T. Cao, S. Shah, C. Maung, and H. Schweitzer, "Cleaning the null space: A privacy mechanism for predictors," in *Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pp. 2789–2795, AAAI, 2017.

[8] X. Liang, T. Zou, B. Guo, S. Li, H. Zhang, S. Zhang, H. Huang, and S. X. Chen, "Assessing beijing's pm2.5 pollution: severity, weather impact, apec and winter heating," in *The Royal Society A*, pp. 471–471, RSPA, 2015.

[9] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, Reis, and J. , "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems, Elsevier*, 2009.

[10] http://mplab.ucsd.edu, "The MPLab GENKI Database, GENKI-4K Subset."

[11] M. Afifi, "Gender recognition and biometric identification using a large dataset of hand images," *arXiv preprint arXiv:1711.04322*, 2017.

[12] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.