ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095632

INFOSHAPE: TASK-BASED NEURAL DATA SHAPING VIA MUTUAL INFORMATION

Homa Esfahanizadeh, William Wu*, Manya Ghobadi, Regina Barzilay, Muriel Médard*

EECS Department, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139 USA

ABSTRACT

The use of mutual information as a tool in private data sharing has remained an open challenge due to the difficulty of its estimation in practice. In this paper, we propose *InfoShape*, a task-based encoder that aims to remove unnecessary sensitive information from training data while maintaining enough relevant information for a particular ML training task. We achieve this goal by utilizing mutual information estimators that are based on neural networks, in order to measure two performance metrics, privacy and utility. Using these together in a Lagrangian optimization, we train a separate neural network as a lossy encoder. We empirically show that *InfoShape* is capable of shaping the encoded samples to be informative for a specific downstream task while eliminating unnecessary sensitive information. Moreover, we demonstrate that the classification accuracy of downstream models has a meaningful connection with our utility and privacy measures.

***Index Terms*—** Task-based encoding, privacy, utility, mutual information, private training.

1. INTRODUCTION

Mutual information (MI) is a measure to quantify how much information is obtained about one random variable by observing another random variable [1]. In a data sharing setting, the data-owner often would like to transform their sensitive samples such that only the necessary information for a specific task is preserved, while sensitive information that can be used for adversarial purposes is eliminated. MI is an excellent candidate that can be used to develop task-based compression for data-sharing to address the privacy-utility trade-off problem [2]. However, estimating MI without knowing the distribution of original data and transformed data is very difficult, and, consequently, using this critical metric has remained limited. In this paper, we utilize numerical estimation of MI to train a task-based lossy encoder for data sharing.

This research was partially sponsored by NSF under grant no. CNS-2008624, Wellcome Trust Fellowship, MIT J-Clinic, the United States Air Force Research Laboratory, and the United States Air Force Artificial Intelligence Accelerator (under Cooperative Agreement Number FA8750- 19-2-1000). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

*These authors contributed equally to this work.

Machine learning (ML) efforts in various sensitive domains face a major bottleneck due to the shortage of publicly available training data [3]. Acquisition and release of sensitive data is a primary issue currently hindering the creation of public large-scale datasets. For example, certain federal regulatory laws such as HIPAA [4] and GDPR [5] prohibit medical centers from sharing their patients' identifiable information. This motivates us to approach the issue from an information theoretic perspective. Our goal is to enable data-owners to eliminate sensitive parts of their data that are not critical for a specific training task before data sharing. We consider a setting where a lossy compressor encodes the data according to two objectives: (i) training a shared model on the combined encoded data of several institutions with a predictive utility that is comparable to the un-encoded baseline; (ii) limiting the use of data for adversarial purposes. In practice, there is a trade-off between the utility and privacy goals.

The state-of-the-art solutions to tackle this privacy-utility trade-off problem mainly involve data-owners sharing their encrypted data, distorted data, or transformed data. Cryptographic methods [6, 7, 8] enable training ML models on encrypted data and offer extremely strong security guarantees. However, these methods have a high computational and communication overhead, thereby hindering practical deployment. Distorting the data by adding noise is another solution which can obtain the theoretical notion of differential privacy [9, 10, 11], but, unfortunately, often results in notable utility cost. Finally, transformation schemes convert the sensitive data from the original representation to an encoded representation by using a randomly-chosen encoder [12, 13, 14]; however, if the instance of the random encoder chosen by the data-owner is revealed, the original data can be reconstructed.

In contrast, we design an encoding scheme to convert the original representation of the training data into a new representation that excludes sensitive information. Thus, the privacy comes from the *lossy* behaviour of the encoder (i.e., compressor) that we design for a targeted training task. The privacy goal is to limit the disclosed information about sensitive features of a sample given its encoded representation, and the utility goal is to obtain a competent classifier when trained on the encoded data. We propose a dual optimization approach to preserve privacy while maintaining utility. In particular, we use MI to quantify the privacy and utility performance, and we train a neural network that plays the role of our lossy encoder.

There has been recent progress for estimating bounds on

the mutual information via numerical methods [15, 16, 17, 18]. We combine the privacy and utility measures using MI estimations into a single loss metric to improve an encoder in its training phase. Once the encoder is trained, it is utilized by individual data owners as a task-based lossy compressor to encode their data for release with the associated labels.

2. PROBLEM STATEMENT

We denote the set of all samples of a distribution by $\mathcal{X}$. Each sample $x \in \mathcal{X}$ is labeled via function $L : \mathcal{X} \to \mathcal{Y}$. A data-owner has a sensitive dataset $\mathcal{D} \subseteq \mathcal{X}$ that she wishes to outsource to a third party for training a specific classification model (i.e., to learn the function L). For privacy concerns, the data-owner first encodes the sensitive data, via an encoder $T : \mathcal{X} \to \mathcal{Z}$, and then publicly releases the labeled encoded data $\{(T(x), L(x))\}_{x\in\mathcal{D}}$. An adversary may have access to the deposited dataset, but uses it for adversarial purposes, i.e., deriving a sensitive feature $S(x)$ from $T(x)$, where $S : \mathcal{X} \to \mathcal{Y}'$. We call $L(x)$ and $S(x)$ the public and private label of sample $x \in \mathcal{X}$, respectively.

The utility goal is to preserve from each sample as much information as needed to train a competitive downstream classification model. The privacy goal is to eliminate unnecessary sensitive data from each sample, which is not critical for the training task but might be misused by an adversary. There are several methods to quantify the privacy and utility performance, and here, we use Shannon entropy [1].

Definition 1. The utility score is negative of the average uncertainty about the public label given its encoded representation,

$$M_{\text{utility}}(T) \triangleq -\,\mathrm{H}[L(x)|T(x)]. \tag{1}$$

There are two potential ways to express the privacy: Given the encoded representation, one is the average uncertainty about the original sample and one is the average uncertainty about a sensitive feature of the original sample. While each can be advantageous over the other depending on the problem setting, without loss of generality and for simplicity, we use the second approach in this paper.

Definition 2. The privacy score is the average uncertainty about the private label given its encoded representation,

$$M_{\text{privacy}}(T) \triangleq \mathrm{H}[S(x)|T(x)]. \tag{2}$$

The privacy and utility are competing targets, and in this paper, we design a lossy encoder that offers a desired trade-off via a Lagrangian optimization. Consider the family of possible encoders as $\mathcal{T}$. An optimal encoder $T^* \in \mathcal{T}$ is obtained as,

$$T^* = \arg\min_{T\in\mathcal{T}} M_{\text{utility}}(T) + \lambda M_{\text{privacy}}(T),$$

where λ is a non-negative metric that controls the trade-off between privacy and utility, chosen to be 1 in our experiment.

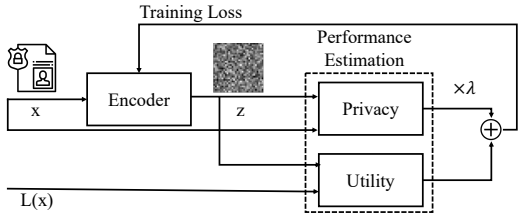

Fig. 1. InfoShape design procedure: At each training iteration, the privacy and utility are scored for improving the encoder.

There has been increasing theoretical interest in using information theoretic measures to encode data for privacy goals. These are organized under the Information Bottleneck method [19]. However, since it is difficult to calculate these measures due to their dependence on certain (often intractable) probability distributions, they have remained impractical to use. Recent efforts for estimating and incorporating these measures have also faced practical challenges, and deriving connections between optimizing variations of these measures and the success of the task-based encoding (both the utility and privacy aspects) are still open challenges [15, 17, 20].

3. ELIMINATING SENSITIVE DATA

We propose a dual optimization mechanism, dubbed *InfoShape*, to simultaneously preserve privacy while also maintaining utility on downstream classification tasks, see Fig. 1. We choose the name of *InfoShape* since our scheme trains a neural network encoder to act as a task-specific lossy compressor, by keeping as much relevant information as possible for our intended downstream task while "shaping" the data to achieve a private representation.

Consider *InfoShape* as an encoder T_θ with set of parameters θ (i.e., an ML model with weights described by θ). We define the loss metric $Q(\theta)$ for evaluating the privacy-utility performance of T_θ as follows:

$$Q(\theta) = M_{\text{privacy}}(T_\theta) + \lambda M_{\text{utility}}(T_\theta). \tag{3}$$

This loss is used for training the encoder by adjusting θ.

Our optimization problem is to determine the set of parameters θ such that the loss metric defined in Eq (3) is minimized. We solve this optimization , i.e., $\theta^* = \arg\min Q(\theta)$, numerically via the stochastic gradient descent (SGD) method [21],

$$G = \nabla Q(\theta^{itr}), \quad \theta^{itr+1} = h\left(\theta^{itr}, G\right). \tag{4}$$

Eq. (4) shows the gradient of the loss function with respect to weights of the encoder, as well as the weight update step. Here, $h(.)$ is a gradient-based optimizer.

Once the encoder is trained, it can be utilized by individual data-owners as a task-based lossy compressor to encode their data and to enable the release of data for collaborative training.

3.1. Neural Estimation of Performance Scores

We utilize neural estimation of MI to numerically approximate the privacy and utility scores. For this, we re-write the privacy

and utility scores in Eq. (1)-(2), as follows:

$$\begin{aligned} M_{\text{utility}}(T) &= \mathrm{I}[L(x);T(x)] - \mathrm{H}[L(x)], \\ M_{\text{privacy}}(T) &= \mathrm{H}[S(x)] - \mathrm{I}[S(x);T(x)]. \end{aligned} \tag{5}$$

Note that the terms $\mathrm{H}[L(x)]$ and $\mathrm{H}[S(x)]$ do not depend on the encoder, and so they vanish in the gradient.

For training the lossy encoder, we use a set of samples $\{x, L(x), S(x)\}_{x\in\mathcal{P}}$, such that $\mathcal{P} \subset \mathcal{X} \setminus \mathcal{D}$. The underlying distributions are unknown (e.g., $\mathrm{P}[L(x)|x]$ which characterizes a perfect classifier, and $\mathrm{P}[S(x)|T(x)]$ which characterizes a computationally unbounded adversary). Consequently, MI is difficult to compute for a finite dataset of high-dimensional inputs [22]. Thus, we consider tractable variational lower bounds that approximate MI [15, 18].

Let us consider two random variables $\alpha \in \mathcal{A}$ and $\beta \in \mathcal{B}$. By definition, MI can be expressed in terms of the KL-divergence (a measure of distance between two distributions [23]) between the joint distribution and multiplications of the marginal distributions of α and β,

$$\mathrm{I}[\alpha;\beta] = \mathrm{D}_{\mathrm{KL}}(\mathrm{P}[\alpha,\beta]||\mathrm{P}[\alpha]\mathrm{P}[\beta]) = \sum_{\alpha,\beta} \mathrm{P}[\alpha,\beta] \log \frac{\mathrm{P}[\alpha,\beta]}{\mathrm{P}[\alpha]\,\mathrm{P}[\beta]}.$$

The Donsker-Varadhan representation of KL-Divergence [24] is as follows: Let Ω be the product sample space of two distributions P_1 and P_2.

$$\mathrm{D}_{\mathrm{KL}}(\mathrm{P}_1 \,||\, \mathrm{P}_2) = \sup_{F:\Omega\to\mathbb{R}} \mathrm{E}_{\mathrm{P}_1}[F] - \log \mathrm{E}_{\mathrm{P}_2}[e^F],$$

where the supremum is taken over all functions F such that the two expectations are finite. In [15], the sufficiently rich set of functions F is modeled with a neural network F_ϕ parameterized with set of weights $\phi \in \Phi$. Let

$$\mathrm{I}_\phi[\alpha;\beta] = \mathrm{E}_{\mathrm{P}[\alpha,\beta]}[F_\phi] - \log \mathrm{E}_{\mathrm{P}[\alpha]\mathrm{P}[\beta]}[e^{F_\phi}]. \tag{6}$$

The optimal parameter $\phi^* \in \Phi$ that maximizes $\mathrm{I}_\phi[\alpha;\beta]$ can be identified via SGD. $\tilde{\mathrm{I}}[\alpha;\beta] = \mathrm{I}_{\phi^*}[\alpha;\beta]$ acts as a lower bound of $\mathrm{I}[\alpha;\beta]$. In practice, the two expectations in (6) are replaced with empirical averages over samples of a minibatch that are drawn according to $\mathrm{P}[\alpha,\beta]$ and $\mathrm{P}[\alpha]\mathrm{P}[\beta]$, respectively.

Numerically solving (6) using SGD to estimate the MI has some practical challenges. In particular, it suffers from bias and a high-variance in the estimation, when MI is large [25]. To remedy this problem, a regularization term was added to the neural estimation of MI to help with stability [18]:

$$\begin{aligned} \mathrm{I}_\phi[\alpha;\beta] = \mathrm{E}_{\mathrm{P}[\alpha,\beta]}[F_\phi] - \log \mathrm{E}_{\mathrm{P}[\alpha]\mathrm{P}[\beta]}[e^{F_\phi}] \\ - 0.1(\log \mathrm{E}_{\mathrm{P}[\alpha]\mathrm{P}[\beta]}[e^{F_\phi}])^2. \end{aligned} \tag{7}$$

The intuition behind the extra regularization term in Eq. (7) is to encourage the optimizer to concentrate on finding one solution in $\{F_\phi : \phi \in \Phi\}$, rather than drifting within a class of equally-behaved functions. To reduce the bias, multiple mini-batches were used to update the MI estimation. For more details we refer the readers to [18], as our core MI estimation method to compute Eq. (5) in our experimental results.

Algorithm 1 Training *InfoShape* (Optimizing θ).

1: **Input:** λ and $\{x, L(x), S(x)\}_{x\in\mathcal{P}}$.
2: Initialize the encoder parameters θ.
3: **repeat**
4: Find $\tilde{\mathrm{I}}[L(x);T_\theta(x)]$ and $\tilde{\mathrm{I}}[S(x);T_\theta(x)]$.
5: Compute $Q(\theta) = M_{\text{privacy}}(T_\theta) + \lambda M_{\text{utility}}(T_\theta)$.
6: Compute $G = \nabla Q(\theta)$ and update $\theta \leftarrow h(\theta, G)$
7: **until** convergence

3.2. Training Procedure of *InfoShape*

We present the training procedure for designing *InfoShape* that keeps the necessary information for learning the function L, but eliminates the sensitive information needed for learning the function S. By training such an encoder, one can ensure that even if an adversary knows the encoder T_θ, they cannot use it to infer sensitive information about the encoded samples. This is because the encoder is not invertible by construction, and even by having T_θ and public samples with disclosed private labels $\{x, S(x)\}_{x\in\mathcal{P}}$, one cannot train a competent classifier that infers sensitive information in the encoded domain. This fact is also supported by our experimental results showing that training a classifier to estimate the private labels in the encoded domain is highly unsuccessful compared to training a classifier to estimate the public labels.[1]

Algorithm 1 shows the step-by-step procedure for training T_θ. The input for the algorithm is the trade-off parameter λ, and a public set of samples with both public and private labels $\{x, L(x), S(x)\}_{x\in\mathcal{P}}$, line 1. We first sample a set of random layer weights θ for the encoder, line 2. Now, starting from the first iteration and until convergence, we iteratively evaluate the performance of T_θ and update its weights accordingly, lines 4-6. In particular, we need to estimate $\mathrm{I}[L(x);T_\theta(x)]$ and $\mathrm{I}[S(x);T_\theta(x)]$ (line 4), for which we use the neural MI estimator in [18, Algorithm 1] due to its numerical stability advantages. We have provided public access to our code and data at https://github.com/billywu1029/infoshape.

4. SIMULATION RESULTS

In this section, we empirically show the potential of our tasked-based lossy encoding scheme through evaluation of its privacy and utility scores. For this purpose, we consider classifiers that are trained and tested on four separate datasets: original data, randomly encoded data, noisy data (by adding independent Gaussian noise per sample), and encoded data using *InfoShape*. For each training dataset, we compare the accuracy between two classifiers: one that identifies the public label (faithful user) and one that identifies the private label (unfaithful user). Further, we show the estimation of privacy score and utility

[1] Our experiments implemented the task-based encoder using a simple neural network. If an adversary obtains a good prediction accuracy for sensitive features, one can use a more complicated encoder architecture or increase the training iterations to kill more sensitive data.

score at different training epochs of our *InfoShape* encoder using the MI estimators.

We utilize a balanced dataset which contains 10,000 samples, each being a vector of size 10 features belonging to one of four different classes.[2] Each sample has 3 primitive features, 2 redundant features, and 5 noisy features. We created 2 clusters of samples per class, and each cluster is constructed as follows: The primitive features are first drawn independently from a standard Gaussian distribution and then randomly linearly combined within each cluster in order to add covariance. The clusters are placed on the vertices of a 3d hypercube with sides of length 2. For each sample, the redundant features are generated as random linear combinations of the primitive features, and the noisy (useless) features are attained using random noise. Samples and the features are then shuffled, and 99% of samples within each cluster are assigned to the same class. In order to obtain both private labels and public labels for our dataset, we represent each class with two bits (i.e., $\{00, 01, 10, 11\}$), whose most significant and least significant bits represent the private and public labels, respectively.

We first describe our *InfoShape* design parameters: Our encoder is modeled with a dense neural network with 10 input nodes, one intermediate layer with 10 nodes, 3 output nodes, and Tanh non-linearity. We use 50 epochs for training, Adam optimizer, and learning rate 1e-3. As for the loss function, we need to estimate $\mathrm{I}[L(x); T(x)]$ and $\mathrm{I}[S(x); T(x)]$, see Eq. (5). We use the procedure in [18], with custom architectures for the neural networks used for MI estimation: a dense neural network with 4 input nodes (3 for the encoded sample and 1 for the label), two intermediate layers with 100 nodes each, 1 output node, and ReLu non-linearity. For the back propagation algorithm to estimate MI, we run 2000 iterations using an Adam optimizer and a learning rate of 1e-4, with a batch size of 2000 for each iteration. To encourage numerical stability, we only update the gradients for the MI estimation after accumulating and averaging gradients for 10 iterations.

Fig. 2 depicts the validation ROC, a graph showing the performance of a classification model at all classification thresholds: the left and right sub-figures show the classification performance for the public label and private label, respectively. The architecture of the classifiers is a dense neural network with 3 inputs nodes, one intermediate layer with 20 nodes, 1 output node with a Sigmoid activation, and ReLU non-linearity between other layers. We use SGD optimizer, batch size 100, learning rate 1e-4, and 50 epochs for training the classifiers, and we use 20% of data for the validation of trained models.

The dashed black line in Fig. 2 represents the accuracy of classifiers trained on the original un-encoded data. The red, green, and blue lines show the accuracy of classifiers trained on randomly-encoded data, and noisy data, and *InfoShape*-encoded data respectively. Compared to the random encoder baseline, our proposed scheme enables a higher accuracy for

[2]We used the sklearn.datasets.make_classification Python library function to generate this random 4-class dataset.

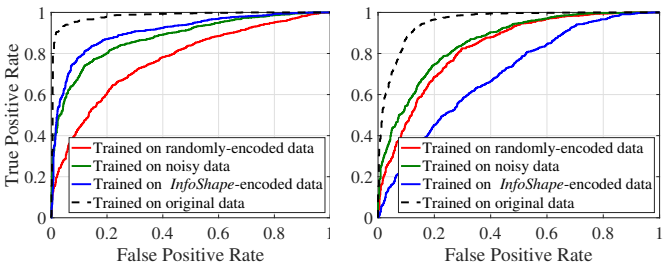

Fig. 2. Validation ROC for modeling (left) public labeling function and (right) sensitive labeling function.

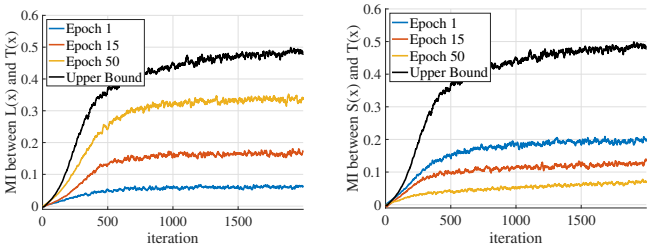

Fig. 3. MI estimation between encoded sample and its public label and private label (left $\mathrm{I}[L(x); T(x)]$; right $\mathrm{I}[S(x); T(x)]$). Graphs are smoothed with a moving average filter. $\mathrm{I}[L(x); x]$ and $\mathrm{I}[S(x); x]$ are shown in black for reference.

the public label and a lower accuracy for the sensitive label, showing success of the task-specific lossy compression. Compared to the Gaussian noise baseline, *InfoShape*-trained model results in slightly better utility performance (AUC[3] of 0.91 vs. 0.88) and much better privacy preservation (AUC 0.69 vs. 0.86), verifying that adding noise indistinguishably degrades the AUC of both the sensitive and public features. This experiment shows that an adversary who has access to some public relevant data and the exact encoder would not achieve a good accuracy to identify the sensitive features of encoded data.

Fig. 3 shows the estimated values of $\mathrm{I}(L(x), T(x))$ and $\mathrm{I}(S(x), T(x))$ per iteration obtained by the MI estimators at various epochs of the encoder training. The average value of the final iterations are used in Eq. (5) to obtain the loss value at each epoch. The estimators show an increase by a factor of 5.1 in the MI between encoded data and public label and a decrease by a factor of 0.65 in the MI between encoded data and sensitive label, between training epochs 1 and 50.

5. CONCLUSION AND FUTURE WORK

In this paper, we presented *InfoShape*, a neural network trained using neural MI estimations to tackle the critical privacy-utility trade-off problem in data sharing. The presented framework can be combined with future MI estimators that offer better numerical stability to apply to real world data, e.g., imaging data in healthcare. The usage of other information measures, such as guesswork, and developing mechanisms to add noise just to the sensitive content of samples remain as future work.

[3]Area under the ROC curve

6. REFERENCES

[1] Claude E. Shannon, "A mathematical theory of communication.," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[2] Ali Makhdoumi, Salman Salamatian, Nadia Fawaz, and Muriel Médard, "From the information bottleneck to the privacy funnel," in *IEEE Information Theory Workshop (ITW)*, 2014, pp. 501–505.

[3] Marzyeh Ghassemi et al., "A review of challenges and opportunities in machine learning for health," in *AMIA Jt Summits Transl Sci*, 2020.

[4] Centers for Medicare & Medicaid Services, "The Health Insurance Portability and Accountability Act of 1996 (HIPAA)," Online at `http://www.cms.hhs.gov/hipaa/`, 1996.

[5] Council of European Union, "Regulation (eu) 2016/679," Online at `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679&qid=1639028015586`, 2016.

[6] Zvika Brakerski and Vinod Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) lwe," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.

[7] Craig Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May*. 2009, pp. 169–178, ACM.

[8] Brandon Reagen et al., "Cheetah: Optimizing and accelerating homomorphic encryption for private inference," in *IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2021, pp. 26–39.

[9] Bo Liu, Ming Ding, Hanyu Xue, Tianqing Zhu, Dayong Ye, Li Song, and Wanlei Zhou, "Dp-image: Differential privacy for image data in feature space," *CoRR*, vol. abs/2103.07073, 2021.

[10] Mohammed Adnan et al., "Federated learning and differential privacy for medical image analysis.," *Nature Scientific Report*, vol. 12, no. 1953, 2022.

[11] Cynthia Dwork, Aaron Roth, et al., "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, , no. 3-4, pp. 211–407, 2014.

[12] Yangsibo Huang, Zhao Song, Kai Li, and Sanjeev Arora, "InstaHide: Instance-hiding schemes for private distributed learning," in *Proceedings of the 37th International Conference on Machine Learning*. 13–18 Jul 2020, vol. 119 of *ML Research*, pp. 4507–4518, PMLR.

[13] Hanshen Xiao and Srinivas Devadas, "Dauntless: Data augmentation and uniform transformation for learning with scalability and security," Cryptology ePrint Archive, Paper 2021/201, 2021, `https://eprint.iacr.org/2021/201`.

[14] Adam Yala, Victor Quach, Homa Esfahanizadeh, Rafael G. L. D'Oliveira, Ken R. Duffy, Muriel Médard, Tommi S. Jaakkola, and Regina Barzilay, "Syfer: Neural obfuscation for private data release," 2022.

[15] Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R. Devon Hjelm, and Aaron C. Courville, "MINE: mutual information neural estimation," *CoRR*, vol. abs/1801.04062, 2018.

[16] Ben Poole, Sherjil Ozair, et al., "On variational bounds of mutual information," *CoRR*, vol. abs/1905.06922, 2019.

[17] Jiaming Song and Stefano Ermon, "Understanding the limitations of variational mutual information estimators," in *International Conference on Learning Representations*, 2020.

[18] Kwanghee Choi and Siyeong Lee, "Regularized mutual information neural estimation," 2021.

[19] Naftali Tishby, Fernando C. Pereira, Nadia Fawaz, and William Bialek, "The information bottleneck method," in *Allerton*, 1999, p. 368–377.

[20] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy, "Deep variational information bottleneck," in *International Conference on Learning Representations*, 2017.

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

[22] Liam Paninski, "Estimation of entropy and mutual information," *Neural Comput.*, vol. 15, no. 6, pp. 1191–1253, jun 2003.

[23] Solomon Kullback, *Information Theory and Statistics*, Wiley, New York, 1959.

[24] M. D. Donsker and S. R. S. Varadhan, "Asymptotic evaluation of certain markov process expectations for large time, i," *Communications on Pure and Applied Mathematics*, vol. 28, no. 1, pp. 1–47, 1975.

[25] David McAllester and Karl Stratos, "Formal limitations on the measurement of mutual information," in *Proc. of the Twenty Third International Conference on Artificial Intelligence and Statistics*. 26–28 Aug 2020, vol. 108 of *ML Research*, pp. 875–884, PMLR.