GENERATIVE ADVERSARIAL USER PRIVACY IN LOSSY SINGLE-SERVER INFORMATION RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of information retrieval from a dataset of files stored on a single server under both a user distortion and a user privacy constraint. Specifically, a user requesting a file from the dataset should be able to reconstruct the requested file with a prescribed distortion, and in addition, the identity of the requested file should be kept private from the server with a prescribed privacy level. The proposed model can be seen as an extension of the well-known concept of private information retrieval by allowing for distortion in the retrieval process and relaxing the perfect privacy requirement. We initiate the study of the tradeoff between download rate, distortion, and user privacy leakage, and show that the optimal rate-distortion-leakage tradeoff is convex and that in the limit of large file sizes this allows for a concise information-theoretical formulation in terms of mutual information. Moreover, we propose a new data-driven framework by leveraging recent advancements in generative adversarial models which allows a user to learn efficient schemes in terms of download rate from the data itself. Learning the scheme is formulated as a constrained minimax game between a user which desires to keep the identity of the requested file private and an adversary that tries to infer which file the user is interested in under a distortion constraint. In general, guaranteeing a certain privacy level leads to a higher rate-distortion tradeoff curve, and hence a sacrifice in either download rate or distortion. We evaluate the performance of the scheme on a synthetic Gaussian dataset as well as on the MNIST dataset. For the MNIST dataset, the data-driven approach significantly outperforms a proposed general achievable scheme combining source coding with the download of multiple files.

1 INTRODUCTION

Machine learning (ML) has been recognized as a game-changer in modern information technology, and various ML techniques are increasingly being utilized for a variety of applications from intrusion detection to image classification and to recommending new movies. Efficient information retrieval (IR) from a single or several servers storing such datasets under a strict user privacy constraint has been extensively studied within the framework of private information retrieval (PIR). In PIR, first introduced by Chor *et al.* (Chor et al., 1995), a user can retrieve an arbitrary file from a dataset without disclosing any information (in an information-theoretic sense) about which file she is interested in to the servers storing the dataset. Typically, the size of the queries is much smaller than the size of a file. Hence, the efficiency of a PIR protocol is usually measured in terms of the download cost, or equivalently, the download (or PIR) rate, neglecting the upload cost of the queries. PIR has been studied extensively over the last decade, see, e.g., (Banawan & Ulukus, 2018; Freij-Hollanti et al., 2017; Kopparty et al., 2011; Sun & Jafar, 2017; Tajeddine et al., 2018; Yekhanin, 2010) and references therein.

Recently, there has been several works proposing to relax the perfect privacy condition of PIR in order to improve on the download cost, see, e.g., (Lin et al., 2020; Samy et al., 2019; Toledo et al., 2016). Inspired by this line of research, we propose to simultaneously relax both the perfect privacy condition and the perfect recovery condition, by allowing for some level of distortion in the recovery process of the requested file, in order to achieve even lower download costs (or, equivalently, higher download rates). We concentrate on the practical scenario in which the dataset is stored on a single server. A problem formulation with arbitrary distortion and leakage functions is presented, which

establishes a tri-fold tradeoff between download rate, privacy leakage to the server storing the dataset, and distortion in the recovery process for the user. We show that the optimal rate-distortion-leakage tradeoff is convex (see Lemma 1) and that it allows for a concise information-theoretical formulation in terms of mutual information in the limit of large file sizes (see Theorem 1). In the special case of full leakage to the server, the proposed formulation yields the well-known rate-distortion curve. The typical behavior of the rate-distortion-leakage tradeoff is illustrated in Fig. 1, showing that an increased level of privacy leads to a higher rate-distortion tradeoff curve, and hence a sacrifice in either download rate or distortion. A general achievable scheme combining source coding with the download of multiple files is proposed for datasets with a known distribution. Moreover, to overcome the practical limitation of unknown statistical properties of real-world datasets, we consider a data-driven approach leveraging recent advancements in generative adversarial networks (GANs) (Goodfellow et al., 2014), which allows a user to learn efficient schemes (in terms of download rate) from the data itself. In our proposed GAN-based framework, learning the scheme can be phrased as a constrained minimax game between a user which desires to keep the identity of the requested file private and a server that tries to infer which file the user is interested in, under both a user distortion and a download rate constraint. Similar to (Springenberg, 2016), where a cross-entropy loss function is used as a discriminative classifier for unlabeled or partially labeled data, the server is modeled as a discriminator in the generalized GAN framework, and also trained with cross-entropy for labeled data. We evaluate the performance of the proposed scheme on a synthetic Gaussian dataset as well as on the MNIST (Lecun et al., 1998) dataset. For the MNIST dataset, the data-driven approach significantly outperforms the proposed achievable scheme, while for the Gaussian dataset, where the source statistics is known, it performs close to the proposed achievable scheme using a variant of the generalized Lloyd's algorithm (Lloyd, 1982; Linde et al., 1980) for the source code.

Related Work

As outlined above, in this work we consider "information retrieval" in the sense of PIR, while "information retrieval" in the traditional sense used by the *information retrieval community* has a different meaning. In particular, in the traditional sense "information retrieval" refers to the problem of providing a list of documents given a query and has a wide range of applications (Baeza-Yates et al., 1999). In Wang et al. (2017), the authors proposed to iteratively optimize two well-established models of traditional information retrieval; namely generative retrieval focusing on predicting relevant documents given a query and discriminative retrieval focusing on predicting document relevance given a query and document pair. The resulting optimization problem is formulated as a minimax game. Due to differences in system model there is no clear connection to our proposed framework, besides the formulation as a minimax game.

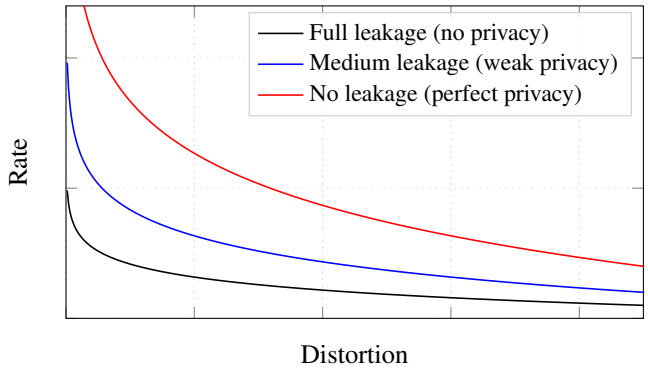


Figure 1: The rate-distortion tradeoff under different privacy levels.

Similar data-driven approaches, under the names of generative adversarial privacy (Huang et al., 2017; 2018), privacy-preserving adverisal networks (Tripathy et al., 2019), and compressive privacy GAN (Tseng & Wu, 2020), have recently been proposed for learning a privatization mechanism directly from the dataset in order to release it to the public and for generating compressed representations that retain utility while being able to withstand reconstruction attacks. A similar approach was also taken in (Blau & Michaeli, 2019) where a tri-fold tradeoff between rate, distortion, and perception in lossy image compression was established. Relaxing the perfect information-theoretical privacy condition by considering computationally-private information retrieval where the privacy requirement relies on an intractability assumption (e.g., the hardness of deciding quadratic residuosity) has been investigated in several previous works, see, e.g., (Kushilevitz & Ostrovsky, 1997; 2000; Lipmaa, 2005). Hence, given infinite computational power, the requested file index can be determined precisely. Moreover, in (Kadhe et al., 2020), it was shown that allowing for side information can also decrease the download cost in single-server PIR. In contrast to these previous works, here we propose

to relax the perfect reconstruction constraint in order to decrease the download cost. Moreover, to the best of our knowledge, addressing PIR or the extension to nonperfect privacy and recovery in the context of generative adversarial models has not been considered in the open literature so far.

2 Preliminaries and System Model

2.1 Notation

We denote $[a] \triangleq \{1, 2, \ldots, a\}$. An arbitrary field is denoted $\mathbb{F}$, while the set of nonnegative real numbers is denoted by $\mathbb{R}_{\geq 0}$. Vectors are denoted by bold letters and sets by calligraphic uppercase letters, e.g., $\boldsymbol{x}$ and $\mathcal{X}$, respectively. We use uppercase letters for random variables (RVs) (either scalar or vector), e.g., X or $\boldsymbol{X}$. For a given index set $[\mathsf{M}]$, we write $X^{[\mathsf{M}]}$ to represent $\left\{X^{(m)} \colon m \in [\mathsf{M}]\right\}$. $\mathsf{E}_X\left[\cdot\right]$ and $\mathsf{E}_{P_X}\left[\cdot\right]$ denote expectation with respect to the RV X and the probability mass function P_X, respectively. $\mathsf{H}\left(X\right)$ or $\mathsf{H}\left(P_X\right)$ represents the entropy of X, while $\mathsf{I}(X\,;Y)$ denotes the mutual information (MI) between X and Y. The multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Σ is denoted as $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$. In particular, if the entries of this distribution are mutually independent, we have $\Sigma = \sigma^2 I$, for marginal standard deviation $\sigma \in \mathbb{R}_{\geq 0}$ and I denoting the identity matrix. The transpose of a vector is denoted as $(\cdot)^{\intercal}$.

2.2 Single-Server Information Retrieval

Consider a dataset containing M files $\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(\mathsf{M})}$ stored on a single server, where each file $\boldsymbol{X}^{(m)} = \left(X_1^{(m)}, \ldots, X_\beta^{(m)}\right)^{\intercal}$, $m \in [\mathsf{M}]$, can be seen as a $\beta \times 1$ random vector (according to some probability distribution $P_{\boldsymbol{X}^{(m)}}$) over $\mathbb{F}^\beta$, where $\mathbb{F}$ is any field (finite or infinite). Assume that the user wishes to retrieve the M-th file, $M \in [\mathsf{M}]$, where, for simplicity, M is assumed to be uniformly distributed over $[\mathsf{M}]$.[1] The formal definition of a general single-server IR scheme is as follows.

Definition 1. *An IR scheme $\mathscr{C}$ for a single-server storing M files consists of: (i) A random strategy $\boldsymbol{S}$. (ii) A deterministic query function f_{Q} that generates a query $\boldsymbol{Q} = f_{\mathsf{Q}}(M, \boldsymbol{S})$, where query $\boldsymbol{Q}$ is sent to the server. (iii) A deterministic answer function f_{A} that returns the answer $\boldsymbol{A} = f_{\mathsf{A}}(\boldsymbol{Q}, \boldsymbol{X}^{[\mathsf{M}]})$ back to the user. (iv) A deterministic reconstruction function $\hat{\boldsymbol{X}} \triangleq f_{\hat{\mathsf{X}}}(\boldsymbol{A}, M, \boldsymbol{Q})$ giving an estimate of the desired file using the answer from the server together with the requested file index M and the query $\boldsymbol{Q}$.*

Note that the server does not use $\boldsymbol{S}$ directly to produce $\boldsymbol{A}$ and thus using $\boldsymbol{Q}$ is sufficient in the reconstruction.

We are interested in designing an IR scheme such that both the user's *utility* and *privacy* are preserved. On the one hand, this scheme should satisfy the condition of retrievability with a distortion measure $d(\cdot, \cdot)$, i.e.,

$$\mathsf{E}_{M,\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(M)}, \hat{\boldsymbol{X}})\right] \leq \mathsf{D}, \tag{1}$$

where D is a given distortion constraint. On the other hand, the user would like to preserve her privacy with the query function, in the sense that the server should not be able to fully determine the identity M of the requested file. The server receives the query $\boldsymbol{Q}$, and the leakage is measured in terms of a leakage metric $\rho(P_{\boldsymbol{Q}|M})$. The query function should be designed such that

$$\rho(P_{\boldsymbol{Q}|M}) \leq \mathsf{L}, \tag{2}$$

where L is the maximum allowed leakage.

Moreover, it is worth mentioning that unlike the setting of classical PIR, where perfect retrievability is ensured for every file, i.e., $\mathsf{E}_{\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(m)}, \hat{\boldsymbol{X}})\right] = 0$ for all $m \in [\mathsf{M}]$, the distortions for different files need not be the same. In other words, it is possible to have $\mathsf{E}_{\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(m)}, \hat{\boldsymbol{X}})\right] \neq \mathsf{E}_{\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(m')}, \hat{\boldsymbol{X}})\right]$ for $m \neq m'$, and (1) can be expressed as

$$\mathsf{E}_{M,\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(M)}, \hat{\boldsymbol{X}})\right] = \mathsf{E}_M\left[\mathsf{E}_{\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(M)}, \hat{\boldsymbol{X}}) \middle| M = m\right]\right] \leq \mathsf{D}.$$

[1] The assumption that M is uniformly distributed can be lifted.

Given a query function f_{Q} and an answer function f_{A} of an IR scheme, we measure its efficiency in terms of the download rate (in bits per symbol) defined as

$$\mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}}) \triangleq \frac{\mathsf{E}_{\boldsymbol{Q},\boldsymbol{A}}\left[\ell_{\boldsymbol{Q}}(\boldsymbol{A})\right]}{\beta} = \frac{1}{\beta}\sum_{\boldsymbol{q}} P_{\boldsymbol{Q}}(\boldsymbol{q})\,\mathsf{E}_{\boldsymbol{A}}\left[\ell_{\boldsymbol{q}}(\boldsymbol{A})\right], \tag{3}$$

where $\ell_{\boldsymbol{q}}(\boldsymbol{a})$ is the length of the answer $\boldsymbol{a}$ for query $\boldsymbol{q}$. Further in the paper, the length $\ell_{\boldsymbol{q}}(\boldsymbol{a})$ is a constant (i.e., independent of $\boldsymbol{a}$ and $\boldsymbol{q}$) that is chosen according to the desired rate.

2.3 Problem Formulation

We consider a model that protects the privacy of the user against a honest-but-curious server. The IR model above guarantees that a user can retrieve an arbitrary file stored on a single server (with some distortion), while the server can only partially infer the identity of the requested file. In this work, we will study the tri-fold tradeoff between download rate, expected distortion, and privacy leakage to the server for an IR scheme. In particular, for a given constraint D on the expected distortion $\mathsf{E}_{M,\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(M)}, f_{\hat{\mathrm{X}}}(\boldsymbol{A}, M, \boldsymbol{Q}))\right]$ for the retrieval of the requested file indexed by M and for a given constraint L on the leakage to the server $\rho(P_{\boldsymbol{Q}|M})$, the goal is to minimize the download rate $\mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}})$. This can be formulated as the constrained optimization problem

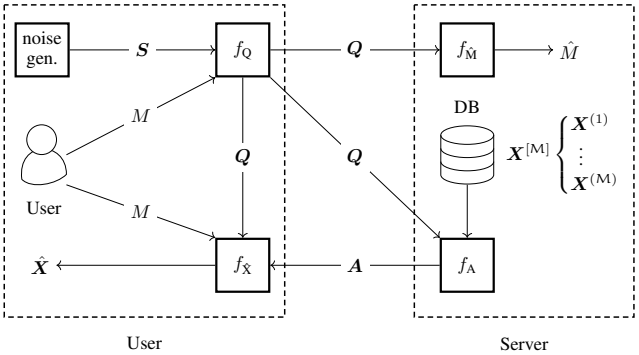


Figure 2: IR scheme for an arbitrary dataset stored on a single server.

$$\min_{f_{\mathrm{Q}}}\min_{f_{\mathrm{A}}} \quad \mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}}) \tag{4a}$$

$$\text{subject to} \quad \mathsf{E}\left[d(\boldsymbol{X}^{(M)}, \hat{\boldsymbol{X}})\right] \leq \mathsf{D}, \quad \rho(P_{\boldsymbol{Q}|M}) \leq \mathsf{L}, \tag{4b}$$

$$\hat{\boldsymbol{X}} = f_{\hat{\mathrm{X}}}(\boldsymbol{A}, M, \boldsymbol{Q}), \quad \boldsymbol{Q} = f_{\mathrm{Q}}(M, \boldsymbol{S}), \quad \boldsymbol{A} = f_{\mathrm{A}}(\boldsymbol{Q}, \boldsymbol{X}^{[\mathrm{M}]}). \tag{4c}$$

To further capture the information leakage for training, we define $\hat{M} \triangleq f_{\hat{\mathrm{M}}}(\boldsymbol{Q})$ to be the server's inference of the user's requested file index M from the query $\boldsymbol{Q}$ using a decision decoder $f_{\hat{\mathrm{M}}}$. The proposed model is illustrated in Fig. 2. The leakage on the identity of the requested file can also be quantified by a loss function denoted as $f_{\mathrm{Loss}}(M, \hat{M})$. Hence, the expected loss with respect to M and $\boldsymbol{Q}$ is

$$\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}}) = \mathsf{E}_{M,\boldsymbol{Q}}\left[f_{\mathrm{Loss}}(M, \hat{M})\right] = \mathsf{E}_{M,\boldsymbol{Q}}\left[f_{\mathrm{Loss}}\big(M, f_{\hat{\mathrm{M}}}(\boldsymbol{Q})\big)\right].$$

Here, the leakage metric $\rho(P_{\boldsymbol{Q}|M})$ in (2) is connected to the expected loss $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$ as $\rho(P_{\boldsymbol{Q}|M}) = \max_{f_{\hat{\mathrm{M}}}} \mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$. Note that the main objective is to minimize the user's leakage to the server, which is equivalent to minimizing $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$, while the goal of the server is the opposite, i.e., of making the user's loss as large as possible or, equivalently, of maximizing $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$.

Lemma 1. *The set of achievable (feasible) rate-distortion-leakage triples* $(\mathsf{R}, \mathsf{D}, \mathsf{L})$ *from* (4) *is a convex set, for any leakage metric* ρ *that is convex in* $P_{\boldsymbol{Q}|M}$.

We remark that Lemma 1 also holds for any loss function f_{Loss}, since any $\rho(P_{\boldsymbol{Q}|M})$ that can be expressed via $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$ is always convex in $P_{\boldsymbol{Q}|M}$.

Note that if $f_{\mathrm{Loss}}(m, \hat{M})$ is the 0-1 loss function (Nguyen & Sanner, 2013) it can be easily shown, following a similar argument as in Huang et al. (2017, Sec. 2.2), that the optimal inference strategy for the server is the MAP decoder, and that the privacy metric $\rho(P_{\boldsymbol{Q}|M})$ in this case equals the server's *inference accuracy* $\Pr(M = \hat{M})$. This case is referred to as hard decision decoding. However, if $f_{\mathrm{Loss}}(m, \hat{M})$ is the log-loss function defined as

$$f_{\mathrm{Loss}}(m, \hat{M}) = \mathsf{H}(M) + \log F_{\hat{M}}(m|\boldsymbol{Q}), \tag{5}$$

then the leakage to the server is measured in terms of MI, i.e., $\rho(P_{\boldsymbol{Q}|M}) = \mathsf{I}(M;\boldsymbol{Q})$. This case is referred to as soft decision decoding. Moreover, the detailed derivation, which can be found in Appendix C, gives an explicit reason (see (C.12)) to use empirical categorical cross-entropy as loss function for the data-driven approach in Section 4. It can be seen that the leakage metrics ρ we considered here are all convex in $P_{\boldsymbol{Q}|M}$. Thus, in the rest of paper, we will always assume that the metric $\rho(\cdot)$ is convex in its argument.

Theorem 1. *The download rate given in* (4) *is equal to the following information rate-distortion-leakage function defined as*

$$\mathsf{R}(\mathsf{D},\mathsf{L}) \triangleq \min_{P_{\boldsymbol{Q}|M}, P_{\hat{\boldsymbol{X}}^{[\mathsf{M}]}|\boldsymbol{X}^{[\mathsf{M}]},\boldsymbol{Q}} \in \mathcal{F}(\mathsf{D},\mathsf{L})} \mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]};\hat{\boldsymbol{X}}^{[\mathsf{M}]} \,\big|\, \boldsymbol{Q}\big), \tag{6}$$

as $\beta \to \infty$*, where*

$$\mathcal{F}(\mathsf{D},\mathsf{L}) \triangleq \left\{ P_{\boldsymbol{Q}|M}, P_{\hat{\boldsymbol{X}}^{[\mathsf{M}]}|\boldsymbol{X}^{[\mathsf{M}]},\boldsymbol{Q}} \colon \mathsf{E}_{M,\boldsymbol{Q}}\big[d(\boldsymbol{X}^{(M)},\hat{\boldsymbol{X}}^{(\mathsf{M})})\big] \le \mathsf{D},\ \rho(P_{\boldsymbol{Q}|M}) \le \mathsf{L} \right\}.$$

The proof that the rate R is bounded from below by $\mathsf{R}(\mathsf{D},\mathsf{L})$, for any β, is given in Appendix B. While the achievability proof is not provided, we do give the following intuition and relative technique from information theory. Note that our single-server IR scheme can be seen as a source coding problem with side information, where the server and the user act as an encoder and a decoder, respectively, and where the query can be seen as a controllable side information known to both the encoder and decoder in order to enforce the privacy condition. This is similar to the well-known Wyner-Ziv problem (Wyner & Ziv, 1976). Hence, a random-coding based scheme can be used to achieve $\mathsf{R}(\mathsf{D},\mathsf{L})$ as $\beta \to \infty$. Further, it can be shown that the alphabet size of the designed queries can be restricted to $\mathsf{M}+4$, which follows from applying the Carathéodory's theorem (Cover & Thomas, 2006, Thm. 15.3.5). Note, however, that finding an analytical closed-form expression for $\mathsf{R}(\mathsf{D},\mathsf{L})$ in our case is intractable even for independent and identically distributed (i.i.d.) files. Hence, we provide numerical results for Gaussian data in Section 5, which indicate that, by combining the achievable scheme proposed below in Section 3 with a convexifying approach, (6) is achievable.

2.4 Generative Adversarial Approach

The inference of the server can also be modeled in a generative adversarial fashion (Huang et al., 2017; 2018; Tripathy et al., 2019; Tseng & Wu, 2020). In particular, the *leakage-distortion* tradeoff for any fixed download rate constraint R is formally described as follows. Consider a family of IR schemes with query generators f_{Q}, answer functions f_{A}, and with download rate $\mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}})$ at most R. The goal of the server is to maximize the expected loss $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$ of the user by designing the decision function $f_{\hat{\mathrm{M}}}$. In contrast, the user would like to design a scheme with f_{Q}, f_{A}, and $f_{\hat{\mathrm{X}}}$ such that the download rate $\mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}}) \le \mathsf{R}$ and such that the expected loss $\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})$ is minimized, while preserving the utility of the scheme, i.e., in the sense that the user can still retrieve the requested file with an expected distortion smaller than a prescribed D. This leads to the constrained minimax optimization problem

$$\min_{f_{\mathrm{Q}}} \max_{f_{\hat{\mathrm{M}}}} \quad \mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}}) \tag{7a}$$

$$\text{subject to} \quad \mathsf{E}_{M,\boldsymbol{Q}}\left[d(\boldsymbol{X}^{(M)},\hat{\boldsymbol{X}})\right] \le \mathsf{D}, \quad \mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}}) \le \mathsf{R}, \tag{7b}$$

$$\hat{\boldsymbol{X}} = f_{\hat{\mathrm{X}}}(\boldsymbol{A}, M, \boldsymbol{Q}), \quad \boldsymbol{Q} = f_{\mathrm{Q}}(M, \boldsymbol{S}), \quad \boldsymbol{A} = f_{\mathrm{A}}(\boldsymbol{Q}, \boldsymbol{X}^{[\mathsf{M}]}). \tag{7c}$$

Note that the minimax formulation in (7a) can be written in a GAN form (Goodfellow et al., 2014) in which $f_{\hat{\mathrm{M}}}$ plays the role of the *discriminator* and f_{Q} plays the role of the *generator*. Thus, the machinery of GANs can be used to determine the leakage-distortion tradeoff. In doing so, (7) is first reformulated as the unconstrained optimization problem

$$\min_{f_{\mathrm{Q}}} \max_{f_{\hat{\mathrm{M}}}} \left[\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}}) + \eta_1\, \mathsf{E}_{M,\boldsymbol{Q}}\big[d(\boldsymbol{X}^{(M)}, f_{\hat{\mathrm{X}}}(\boldsymbol{A}, M, \boldsymbol{Q}))\big] + \eta_2 \mathsf{R}(f_{\mathrm{Q}}, f_{\mathrm{A}}) \right], \tag{8}$$

where η_1 and η_2 are tuning parameters, and $\boldsymbol{Q}$ and $\boldsymbol{A}$ are according to Definition 1. The minimax game in (8) with soft decision decoding, i.e., with the log-loss function in (C.11), will be the basis for training, as described below in Section 4.

3 ACHIEVABLE SCHEMES

In this section, we present a general achievable scheme for an arbitrary number of files M and hard decision leakage (or accuracy) $\mathsf{L} \in \{1, 1/2, 1/3, \ldots, 1/\mathsf{M}\}$.

The construction is based on the following fact. If $\mathsf{L} = 1$, then the problem becomes essentially the problem of lossy compression: the user explicitly tells the server what she needs to download, and the server sends the requested data compressed with some pre-agreed method. One well-known example of such compression is quantization. This scheme is designed against a maximum-likelihood decoder that provides the best inference for the server. Assume that the user wishes to retrieve the M-th file and to ensure leakage of $\mathsf{L} = 1/\mathsf{N}$, $1 \leq \mathsf{N} \leq \mathsf{M}$. We describe the steps of the scheme as follows.

- The user first selects a lossy source coding scheme $\mathcal{C}$ to encode together N files of size β each. Assume that the source coding scheme achieves an average compression size (in bits) of $\log_2 |\mathcal{C}|$ and expected normalized distortion D (normalized by $\mathsf{N}\beta$), respectively.
- The user's query is designed to exactly request N files with indices $M_1, \ldots, M_\mathsf{N}$ such that $M \in \{M_1, \ldots, M_\mathsf{N}\}$, where the $\mathsf{N} - 1$ nondesired indices are chosen uniformly at random. These $\mathsf{N} - 1$ file indices are added in order to "trick" the server and hide the real file index M of interest.
- After receiving the queries sent by the user, the server compresses all N files by using the source coding scheme $\mathcal{C}$, and transmits the answer back to the user.
- The user decompresses the answer using the selected source coding scheme. The $\mathsf{N} - 1$ nondesired reconstructed files are discarded by the user, while the desired reconstructed file is kept.

The rate of the corresponding IR scheme is $\mathsf{R} = \log_2 |\mathcal{C}|/\beta$, while the expected distortion is D and the leakage is L. The performance of this scheme is strongly dependent on the used source coding scheme $\mathcal{C}$. Using Lemma 1, a scheme for any leakage $1/\mathsf{M} \leq \mathsf{L} \leq 1$ (not only reciprocal of an integer) can be constructed.

From the general scheme above, two specific schemes can be constructed based on the source coding scheme used. The first scheme is constructed by selecting a source coding scheme based on quantization. In particular, we use a variant of the generalized Lloyd's algorithm (Lloyd, 1982; Linde et al., 1980). In the sequel, we will refer to this scheme as the compression-based scheme. The second scheme, referred to in the sequel as Shannon's scheme, assumes $\beta \to \infty$ and uses the well-known rate-distortion function from information theory for the source coding scheme (Gray, 1973). In Fig. 3(a) below, together with a convexifying approach described in Appendix D.2, we plot the performance of both schemes for an i.i.d. Gaussian dataset. It is worth mentioning here that by numerically solve (6) for Gaussian data, the lower bound and the approximation values of Shannon's scheme are actually quite close, which shows that the converse bound is achievable for the i.i.d. Gaussian case.

4 DATA-DRIVEN APPROACH

In this section, we describe in detail our proposed data-driven framework for constructing an efficient IR scheme for downloading an arbitrary file from an arbitrary dataset stored on a single server. The four functions in Fig. 2 are represented as deep neural networks and we assume, for now, that they have already been trained.

The user wishes to retrieve the M-th file $\boldsymbol{X}^{(M)}$ and encodes M as a one-hot $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_\mathsf{M}) \in \{0, 1\}^\mathsf{M}$, where $Y_j = 1$ if $j = M$, and $Y_j = 0$ otherwise. Next, the user generates a "noise" vector $\boldsymbol{S} = (S_1, S_2, \ldots, S_\mathsf{M})$, where $S_1, S_2, \ldots, S_\mathsf{M}$ are i.i.d. according to the standard Gaussian distribution $\mathcal{N}(0, 1)$. The concatenation $(S_1, S_2, \ldots, S_\mathsf{M}, Y_1, Y_2, \ldots, Y_\mathsf{M})$ is the input to a deep neural network, representing the function f_Q, for query generation. This network produces the query $\boldsymbol{Q}$ that is sent to the server. The intuition behind this neural network is to hide the value of M. The server's answer is produced by concatenating the stored data $\boldsymbol{X}^{[\mathsf{M}]}$ and the received query $\boldsymbol{Q}$ and then feed the result into a deep neural network, representing the function f_A, for answer construction. The deep neural network produces the answer vector $\boldsymbol{A}$ that is sent back to the user. The user then concatenates $\boldsymbol{A}$ and $(S_1, \ldots, S_\mathsf{M}, Y_1, \ldots, Y_\mathsf{M})$ and feeds the result into a deep neural network for decoding, representing the function $f_{\hat{\mathsf{X}}}$, to produce the estimate $\hat{\boldsymbol{X}}$ of the requested file $\boldsymbol{X}^{(M)}$.

On the server side, a deep neural network, representing the function $f_{\hat{\mathsf{M}}}$, is used to guess the identity of the requested file. The input to the network is the query vector $\boldsymbol{Q}$ and the output (using softmax) is a distribution-like vector $\boldsymbol{F} = (F_1, F_2, \ldots, F_{\mathsf{M}})$, where $\sum_{j=1}^{\mathsf{M}} F_j = 1$, $0 \leq F_j \leq 1$, and where F_j can be interpreted as "with probability/likelihood F_j, the user's requested file index M is equal to j." The server's estimate of the user requested file index is then $\hat{M} = \arg\max_{j \in [\mathsf{M}]} F_j$.

4.1 Learning Algorithm

Let $\boldsymbol{X}^{(M_l)} = \left(X_1^{(M_l)}, \ldots, X_\beta^{(M_l)}\right)$, $l \in [\mathsf{n}]$, denote the l-th requested file during training and $\hat{\boldsymbol{X}}_l$ the corresponding user reconstructed estimate, where n is the number of training samples. Training the deep neural networks representing the functions f_{Q}, f_{A}, $f_{\hat{\mathsf{X}}}$, and $f_{\hat{\mathsf{M}}}$ is done following (C.12) and (8) by first fixing the download rate R and then solving the minimax optimization problem

$$\min_{(f_{\mathsf{Q}}, f_{\mathsf{A}}, f_{\hat{\mathsf{X}}})} \max_{f_{\hat{\mathsf{M}}}} \frac{1}{\mathsf{n}} \sum_{l=1}^{\mathsf{n}} \left[-f_{\text{XE-Loss}}(\boldsymbol{Y}_l, \boldsymbol{F}_l) + \eta \cdot d\left(\boldsymbol{X}^{(M_l)}, \hat{\boldsymbol{X}}_l\right) \right], \tag{9}$$

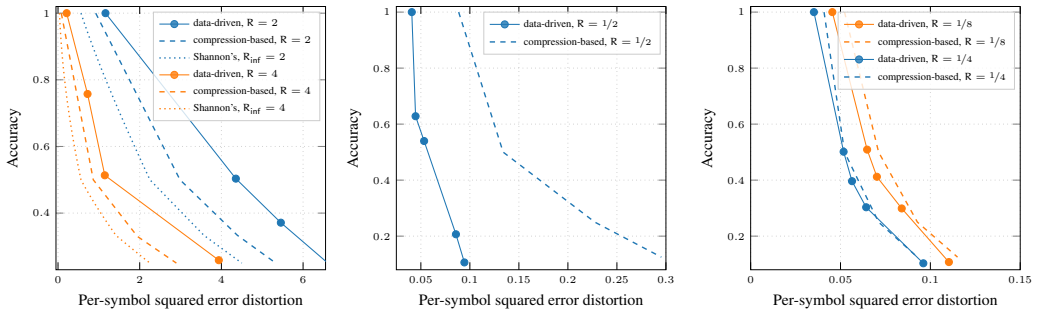
where $f_{\text{XE-Loss}}(\boldsymbol{Y}_l, \boldsymbol{F}_l) = -\sum_{j=1}^{\mathsf{M}} Y_{l,j} \log F_{l,j} = -\log F_{l,M_l}$ measures the categorical cross-entropy between the vectors $\boldsymbol{Y}_l = (Y_{l,1}, \ldots, Y_{l,\mathsf{M}})$ and $\boldsymbol{F}_l = (F_{l,1}, \ldots, F_{l,\mathsf{M}})$. Here, $\boldsymbol{Y}_l$ and $\boldsymbol{F}_l$ is the $\boldsymbol{Y}$-vector and $\boldsymbol{F}$-vector for the l-th sample. The parameter η is a tradeoff coefficient that is increased in every epoch. The solution to the minimax optimization problem in (9) is found using an iterative algorithm employing stochastic gradient decent similar to Algorithm 1 in Huang et al. (2017). The solution is found by first maximizing by the objective function of (9) to determine the optimal $f_{\hat{\mathsf{M}}}$ for a fixed initial triple $(f_{\mathsf{Q}}, f_{\mathsf{A}}, f_{\hat{\mathsf{X}}})$. Next, the optimal triple $(f_{\mathsf{Q}}, f_{\mathsf{A}}, f_{\hat{\mathsf{X}}})$ is found for the given $f_{\hat{\mathsf{M}}}$ from the previous step by minimizing the same objective function. This iterative process is continued until convergence or the maximum number of iterations is exceeded. Note that, although training is done based on a log-loss function (see (C.11)), we evaluate the performance more intuitively using accuracy, i.e., $\Pr(M = \hat{M})$, in Section 5 below.

5 Numerical Results

We demonstrate the application of our proposed data-driven approach to a synthetic Gaussian dataset and also for the MNIST dataset, showing that guaranteeing a certain privacy level leads to a higher rate-distortion tradeoff curve, and hence a sacrifice in either download rate or distortion. We also compare the data-driven approach with both the compression-based scheme (for all datasets) and Shannon's scheme of Section 3 (for the Gaussian dataset). The Gaussian dataset consists of $\mathsf{M} = 4$ files, each of dimension $\beta = 3$ and drawn independently according to $\mathcal{N}(\boldsymbol{\mu}^{(m)}, \sigma^2 I)$, where $\boldsymbol{\mu}^{(1)} = (-3.0, -3.0, 3.0)^\intercal$, $\boldsymbol{\mu}^{(2)} = (3.0, 3.0, 3.0)^\intercal$, $\boldsymbol{\mu}^{(3)} = (3.0, -3.0, -3.0)^\intercal$, $\boldsymbol{\mu}^{(4)} = (-3.0, 3.0, -3.0)^\intercal$, and $\sigma = 3$. For MNIST, the training set is comprised of $\mathsf{n} = 6000$ and size 28×28 images for each digit, and testing is performed on randomly chosen images from 10000 test images. The distortion between a requested file $\boldsymbol{X}^{(m)}$ and its estimate $\hat{\boldsymbol{X}}$ is measured as the per-symbol squared error. The CIFAR-10 dataset consists of $\mathsf{n} = 5000$ and size 32×32 pixels 3 channels training images for each classes. There are 10 classes in CIFAR-10 dataset and there are 1000 testing images for each class.

In Fig. 3, we plot the accuracy, or $\Pr(M = \hat{M})$, as function of per-symbol squared error distortion for different download rates R for the data-driven approach (solid curves) and the compression-based scheme (dashed curves). For MNIST a symbol is a grey-scale pixel, while for the Gaussian dataset it is a one-dimensional Gaussian RV. As a comparison, for the Gaussian dataset, we also plot the performance of Shannon's scheme (dotted curves), assuming $\beta \to \infty$, as outlined in Section 3. For Shannon's scheme, we use the *information rate*, denoted by $\mathsf{R}_{\mathsf{inf}}$ and defined as $\mathsf{H}(\boldsymbol{A} \,|\, \boldsymbol{Q})/\beta$. It is well-known that the information rate is a true lower bound on the *operational* rate from (3) (cf. Cover & Thomas (2006)). Although $\mathsf{R}_{\mathsf{inf}}$ is a true lower bound on R, their numerical values appear to be close, e.g., for the Gaussian dataset and $\mathsf{D} = 4.35$, $\mathsf{R} = 2$ while $\mathsf{R}_{\mathsf{inf}} \approx 1.97$.

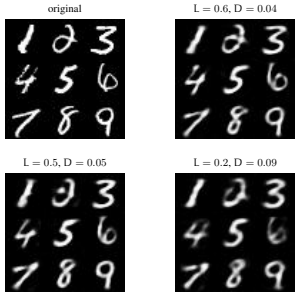
As expected, one can have a higher privacy level (i.e., smaller leakage) for a given distortion at the expense of a higher download rate. For the MNIST dataset, the data-driven approach significantly outperforms the compression-based scheme, while for the Gaussian dataset it performs close to the compression-based scheme using a variant of the generalized Lloyd's algorithm (Lloyd, 1982; Linde et al., 1980) for the source code. This should not come as a surprise, as for the Gaussian dataset,



(a) (b) (c)

Figure 3: Accuracy versus per-symbol squared error distortion for both the data-driven approach and the schemes from Section 3. (a) Synthetic Gaussian dataset. (b) MNIST. (c) CIFAR-10.

the probabilistic model is simple and known precisely. In particular, we believe that the generalized Lloyd's algorithm gives close-to-optimal values (cf. Sabin & Gray (1986)). For MNIST we combined JPEG-like compression (including discrete-cosine transform), run-length encoding, and other entropy coding techniques for the compression-based scheme (cf. Bocharova (2010, Sec. 8.2)). However, due to a very small size of images, the constant overhead (e.g., for storing the Huffman codebook) turned out to be unacceptably high. It is a well-known phenomenon that sophisticated compression methods work well only on files of medium and large size. We thus opted for scalar quantization (see Appendix D.3 for details). Below, we also show the reconstructed digits for three levels of distortion and accuracy for a download rate of $R = 1/2$ for the MNIST dataset.



So far only the download cost of the proposed schemes has been considered, neglecting both the query upload cost and the cost of distributing the trained neural networks. First, the cost of distributing the trained networks can be neglected since training is usually done beforehand on a dedicated training server and can hence be seen as a one-time cost. This cost vanishes as the protocol can in principle run for a very long time serving a large number of users while the dataset grows continuously. Second, the query upload cost is in most cases much smaller than the download cost of the answers. As we will show below, this is also the case here for MNIST. For completeness, the one-time cost (in bits) of distributing the networks to the user is 465479×32 and 1259135×32 bits for the Gaussian and MNIST datasets, respectively, while the corresponding one-time cost of distributing the answer generation network to the server is 532230×32 and 7336888×32 bits, respectively.

	Download cost (R)	Upload cost	Download dataset
Gaussian	$6\,(2), 12\,(4)$	128	384
MNIST	$392\,(1/2)$	320	17825

The number of neurons of the output layer of the query network is always equal to the number of files M. Thus, the upload cost is at most $32 \cdot M$ (assuming 32-bits floats). For MNIST, this yields $32 \cdot 10 = 320$ bits, while the download cost is $28 \cdot 28 \cdot 1/2 = 392$ bits (assuming $R = 1/2$), which is higher. Downloading the entire dataset requires an overall communication cost of $28 \cdot 28 \cdot 8 \cdot 10 = 62720$ bits uncompressed, which is significantly higher. Moreover, the average losslessly compressed sizes for different digits range from 1351 (digit 1) to 1988 (digit 0) and leads to a total average size of 17825 bits, which is again significantly higher than the download cost. In general, the upload cost scales linearly with the number of files and is independent of the file size, while the download cost increases with the file size. In general, the file size is much larger than the number of files which is the standard argument for not considering the upload cost. In the embedded table we summarize the different costs (in bits) for both the Gaussian and MNIST datasets and compare with the cost (in bits) of downloading the entire dataset using lossless source coding.

Note that regarding the problem of overfitting, we have shown an example of learning curves for the CIFAR-10 dataset in Fig. 4. The training distortion curve and training adversary accuracy curve are smoothed by a uniform filter with size 100. The testing distortion and testing adversary accuracy are sampled every 5000 epochs. The testing distortion is decreasing and close to training distortion curve; so the overfitting is negligible. Similarly, training and testing accuracy values are also close.

On the other hand, one might wonder whether we have mode-collapse problem in our case. Mode-collapse means that generator find some weak point of discriminator and keep producing this mode of output to trick discriminator. This may happen when the generator try to fool discriminator trained on the real dataset and cause problem because this kind of GAN is required to have diversity in its output. However, our generator produces query which is not real in world, and mode collapse is not a problem for our scheme because we do not aim to have diversity in query generation. We agree that some technique to prevent GAN mode collapse may be applied to our training such as minibatch discrimination (Salimans et al., 2016). However, preventing mode collapse in GAN is out of scope of this paper. To show that our generative adversarial training is close to optimal, we have shown the MAP accuracy compared to the adversary accuracy for Gaussian case and it shows that our adversary is trained quite well (see the last two sub-tables of Table 2 in Appendix E). Here, we also add the corresponding MAP accuracy for the training result of CIFAR-10 with accuracy $\mathrm{L} = 0.396$ and distortion $\mathrm{D} = 0.056$ in Fig. 4. The MAP accuracy is $\mathrm{L}_{\mathsf{MAP}} = 0.410701875$ by sampling 40000000 points.

6 Conclusion

In this work, we have studied the tradeoff between download rate, privacy leakage to the server, and reconstruction distortion at the user for single-server IR schemes. An IR scheme can be seen as an extension of the well-known concept of PIR by allowing for distortion in the retrieval process and relaxing the perfect privacy requirement. An information-theoretic formulation for the tradeoff in terms of mutual information has been provided in the limit of a large file size. We have shown that generative adversarial models can be successfully applied to design efficient single-server IR schemes. This is in particular beneficial if the data statistics is unknown. The main ingredient is a new optimization approach which combines GANs with additional constraints for the download rate and the desired reconstruction distortion at the user. We have shown that our proposed GAN-based data-driven approach for a fixed download rate is able to provide a tradeoff between distortion for the user and privacy leakage to the server, which is close to that of a proposed compression-based scheme where the source statistics is known. A similar tradeoff as for the Gaussian case can be observed if the proposed data-driven approach is applied to a real-world dataset like MNIST, for which it significantly outperforms the compression-based scheme.

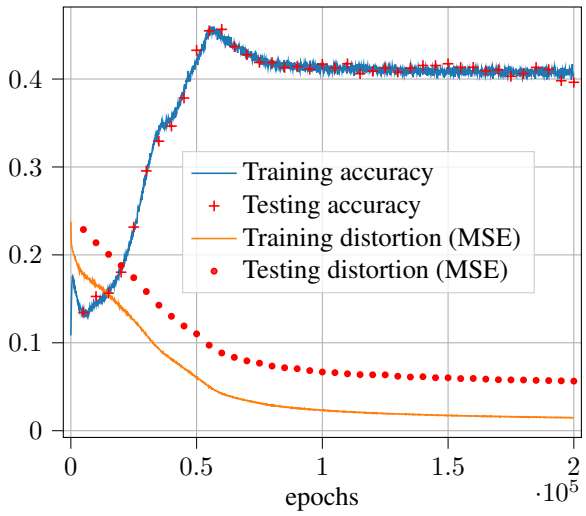


Figure 4: The learning curve of CIFAR-10 achieving leakage $\mathrm{L} = 0.396$, distortion $\mathrm{D} = 0.056$, and rate $\mathrm{R} = 1/4$.

References

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern Information Retrieval*, volume 463. ACM press New York, 1999.

Karim Banawan and Sennur Ulukus. The capacity of private information retrieval from coded databases. *IEEE Transactions on Information Theory*, 64(3):1945–1956, March 2018. ISSN

0018-9448. doi: 10.1109/TIT.2018.2791994.

Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *Proceedings International Conference on Machine Learning (ICML)*, Long Beach, CA, USA, June 9–15, 2019. PMLR.

Irina Bocharova. *Compression for multimedia*. Cambridge University Press, Cambridge, UK, 2010.

Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *Proceedings 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 41–50, Milwaukee, WI, USA, October 23–25, 1995. doi: 10.1109/SFCS.1995.492461.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, NY, USA, second edition, 2006.

Ragnar Freij-Hollanti, Oliver W. Gnilke, Camilla Hollanti, and David A. Karpuk. Private information retrieval from coded databases with colluding servers. *SIAM Journal on Applied Algebra and Geometry*, 1(1):647–664, November 2017. doi: 10.1137/16M1102562.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings 28th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, Montreal, Canada, December 8–13, 2014.

Robert M. Gray. A new class of lower bounds to information rates of stationary sources via conditional rate-distortion functions. *IEEE Transactions on Information Theory*, 19(4):480–489, July 1973. ISSN 1557-9654. doi: 10.1109/TIT.1973.1055050.

Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. Context-aware generative adversarial privacy. *Entropy*, 19(12), December 2017. ISSN 1099-4300. doi: 10.3390/e19120656.

Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. Generative adversarial privacy. In *Proceedings ICML'18 Workshop: Privacy in Machine Learning and Artificial Intelligence*, Stockholm, Sweden, July 15, 2018.

Swanand Kadhe, Brenden Garcia, Anoosheh Heidarzadeh, Salim El Rouayheb, and Alex Sprintson. Private information retrieval with side information. *IEEE Transactions on Information Theory*, 66 (4):2032–2043, April 2020. ISSN 1557-9654. doi: 10.1109/TIT.2019.2948845.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, April 14–16, 2014.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Proceedings 31th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 972–981, Long Beach, CA, USA, December 4–9, 2017.

Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. In *Proceedings 43rd Annual ACM Symposium on Theory of Computing*, pp. 167–176, San Jose, CA, USA, June 06–08, 2011. ISBN 978-1-4503-0691-1. doi: 10.1145/1993636.1993660.

Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: single database, computationally-private information retrieval. In *Proceedings 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 364–373, October 1997. doi: 10.1109/SFCS.1997.646125.

Eyal Kushilevitz and Rafail Ostrovsky. One-way trapdoor permutations are sufficient for non-trivial single-server private information retrieval. In *Proceedings 19th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pp. 104–121, Bruges, Belgium, May 14–18, 2000. ISBN 978-3-540-45539-4.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

Clément Levrard. Quantization/clustering: when and why does k-means work? arXiv:1801.03742v2 [math.ST], January 2018. URL https://arxiv.org/abs/1801.03742.

Hsuan-Yin Lin, Siddhartha Kumar, Eirik Rosnes, Alexandre Graell i Amat, and Eitan Yaakobi. The capacity of single-server weakly-private information retrieval. arXiv:2001.08727v2 [cs.IT], July 2020. URL https://arxiv.org/abs/2001.08727.

Yoseph Linde, Andrés Buzo, and Robert M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, January 1980.

Helger Lipmaa. An oblivious transfer protocol with log-squared communication. In *Proceedings 8th International Conference on Information Security (ISC)*, pp. 314–328, Singapore, September 20–23, 2005. ISBN 978-3-540-31930-6.

Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2):129–137, March 1982.

Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *Proceedings IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4394–4402, Salt Lake City, UT, USA, June 18–23, 2018. doi: 10.1109/CVPR.2018.00462.

Tan T. Nguyen and Scott Sanner. Algorithms for direct 0-1 loss optimization in binary classification. In *Proceedings International Conference on Machine Learning (ICML)*, pp. III–1085–III–1093, Atlanta, GA, USA, June 16–21, 2013.

Michael John Sabin and Robert M. Gray. Global convergence and empirical consistency of the generalized Lloyd algorithm. *IEEE Transactions on Information Theory*, 32(2):148–155, March 1986. ISSN 1557-9654. doi: 10.1109/TIT.1986.1057168.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Proceedings 30th International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2234–2242, Barcelona, Spain, December 5–10, 2016.

Islam Samy, Ravi Tandon, and Loukas Lazos. On the capacity of leaky private information retrieval. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 1262–1266, Paris, France, July 07–12, 2019.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *Proceedings International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, May 2–4, 2016.

Hua Sun and Syed Ali Jafar. The capacity of private information retrieval. *IEEE Transactions on Information Theory*, 63(7):4075–4088, July 2017. ISSN 0018-9448. doi: 10.1109/TIT.2017.2689028.

Razane Tajeddine, Oliver W. Gnilke, and Salim El Rouayheb. Private information retrieval from MDS coded data in distributed storage systems. *IEEE Transactions on Information Theory*, 64 (11):7081–7093, November 2018. ISSN 0018-9448. doi: 10.1109/TIT.2018.2815607.

Raphael R. Toledo, George Danezis, and Ian Goldberg. Lower-cost ϵ-private information retrieval. In *Proceedings Privacy Enhancing Technologies Symposium (PETS)*, pp. 184–201, Darmstadt, Germany, July 19–22, 2016.

Ardhendu Tripathy, Ye Wang, and Prakash Ishwar. Privacy-preserving adversarial networks. In *Proceedings 57th Allerton Conference on Communication, Control, and Computing*, pp. 495–505, Monticello, IL, USA, September 24–27, 2019. doi: 10.1109/ALLERTON.2019.8919758.

Bo-Wei Tseng and Pei-Yuan Wu. Compressive privacy generative adversarial network. *IEEE Transactions on Information Forensics and Security*, 15:2499–2513, 2020.

Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. IRGAN: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings 40th International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 515–524, Shinjuku, Tokyo, Japan, August 7–11, 2017, 2017. ISBN 9781450350228. doi: 10.1145/3077136.3080786.

Aaron D. Wyner and Jacob Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1):1–10, January 1976.

Sergey Yekhanin. Private information retrieval. *Commun. ACM*, 53(4):68–73, April 2010. ISSN 0001-0782. doi: 10.1145/1721654.1721674.

Appendix

A Proof of Lemma 1

Assume the schemes that achieve the triples $(\mathsf{R}_1, \mathsf{D}_1, \mathsf{L}_1)$ and $(\mathsf{R}_2, \mathsf{D}_2, \mathsf{L}_2)$ are $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. We denote the queries of $\mathcal{C}_1$ and $\mathcal{C}_2$ as Q_1 and Q_2, respectively. From $\mathcal{C}_1$ and $\mathcal{C}_2$, we construct a new scheme $\mathcal{C}_\lambda$ that achieves the triple $(\lambda\mathsf{R}_1 + (1-\lambda)\mathsf{R}_2, \lambda\mathsf{D}_1 + (1-\lambda)\mathsf{D}_2, \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2)$, for any $\lambda \in (0,1)$. This is exactly the definition of a convex set. The scheme $\mathcal{C}$ is as follows:

- The user generates a time-sharing RV K according to the distribution

$$P_K(k) = \begin{cases} \lambda & \text{if } k = 1, \\ 1-\lambda & \text{if } k = 2. \end{cases}$$

- Then, the user forms the query $Q = (K, Q_K)$ and sends it to the server, thus explicitly notifying which of the schemes, $\mathcal{C}_1$ or $\mathcal{C}_2$, is used.
- Finally, the server replies according to $\mathcal{C}_K$ using the query Q_K.

The leakage of $\mathcal{C}_\lambda$ becomes

$$\rho(P_{\boldsymbol{Q}|M}) = \rho(\lambda P_{\boldsymbol{Q}_1|M} + (1-\lambda)P_{\boldsymbol{Q}_2|M}) \overset{(a)}{\leq} \lambda\rho(P_{\boldsymbol{Q}_1|M}) + (1-\lambda)\rho(P_{\boldsymbol{Q}_2|M}) = \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2,$$

where equality in (a) can be achieved by artificially "worsening" the scheme $\mathcal{C}_\lambda$, e.g., by explicitly sending M to the server in some cases (with some probability). The proof for the distortion and the rate of $\mathcal{C}$ follows in a similar manner (they are not just convex but also linear). Hence, the set of achievable rate-distortion-leakage triples constitutes a convex set.

B Proof of Theorem 1

Convexity. We prove the convexity for $\mathsf{R}(\mathsf{D}, \mathsf{L})$ based on the assumption that $\rho(P_{\boldsymbol{Q}|M})$ is convex in $P_{\boldsymbol{Q}|M}$, i.e., we have

$$\begin{aligned} \mathsf{L}_\lambda &\triangleq \rho(\lambda P_{\boldsymbol{Q}_1|M} + (1-\lambda)P_{\boldsymbol{Q}_2|M}) \\ &\leq \lambda\rho(P_{\boldsymbol{Q}_1|M}) + (1-\lambda)\rho(P_{\boldsymbol{Q}_2|M}) \leq \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2, \quad 0 \leq \lambda \leq 1. \end{aligned}$$

Note that since the queries should be generated without knowing any realizations of the retrieved files, it is quite natural to have $\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \boldsymbol{Q}\big) = 0$. Thus, the objective function in the minimization of (6) can be expressed as

$$\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}^{[\mathsf{M}]} \,|\, \boldsymbol{Q}\big) = \mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}^{[\mathsf{M}]} \,|\, \boldsymbol{Q}\big) + \underbrace{\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \boldsymbol{Q}\big)}_{=0} = \mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}^{[\mathsf{M}]}, \boldsymbol{Q}\big).$$

Now, let the distributions $P_{\hat{\boldsymbol{X}}_1^{[\mathsf{M}]}, \boldsymbol{Q}_1|\boldsymbol{X}^{[\mathsf{M}]}}$ and $P_{\hat{\boldsymbol{X}}_2^{[\mathsf{M}]}, \boldsymbol{Q}_2|\boldsymbol{X}^{[\mathsf{M}]}}$ achieve $\mathsf{R}(\mathsf{D}_1, \mathsf{L}_1)$ and $\mathsf{R}(\mathsf{D}_2, \mathsf{L}_2)$, respectively. Given $P_{\hat{\boldsymbol{X}}_\lambda^{[\mathsf{M}]}, \boldsymbol{Q}_\lambda|\boldsymbol{X}^{[\mathsf{M}]}} = \lambda P_{\hat{\boldsymbol{X}}_1^{[\mathsf{M}]}, \boldsymbol{Q}_1|\boldsymbol{X}_1^{[\mathsf{M}]}} + (1-\lambda)P_{\hat{\boldsymbol{X}}_2^{[\mathsf{M}]}, \boldsymbol{Q}_2|\boldsymbol{X}^{[\mathsf{M}]}}$ for any $0 \leq \lambda \leq 1$, using the convexity of mutual information in $P_{\hat{\boldsymbol{X}}^{[\mathsf{M}]}, \boldsymbol{Q}|\boldsymbol{X}^{[\mathsf{M}]}}$ we obtain

$$\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}_\lambda^{[\mathsf{M}]}, \boldsymbol{Q}_\lambda\big) \leq \lambda\,\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}_1^{[\mathsf{M}]}, \boldsymbol{Q}_1\big) + (1-\lambda)\,\mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}_2^{[\mathsf{M}]}, \boldsymbol{Q}_2\big). \tag{B.10}$$

Further, one can also see that $P_{\boldsymbol{Q}_\lambda|M} = \lambda P_{\boldsymbol{Q}_1|M} + (1-\lambda)P_{\boldsymbol{Q}_2|M}$ and $P_{\hat{\boldsymbol{X}}_\lambda^{[\mathsf{M}]}|\boldsymbol{X}^{[\mathsf{M}]}} = \lambda P_{\hat{\boldsymbol{X}}_1^{[\mathsf{M}]}|\boldsymbol{X}^{[\mathsf{M}]}} + (1-\lambda)P_{\hat{\boldsymbol{X}}_2^{[\mathsf{M}]}|\boldsymbol{X}^{[\mathsf{M}]}}$. From the convexity of $\rho(\cdot)$ and the linearity of the distortion function $d(\cdot,\cdot)$, it is straightforward to see that $\lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2 \geq \mathsf{L}_\lambda$ and $\lambda\mathsf{D}_1 + (1-\lambda)\mathsf{D}_2 \geq \mathsf{D}_\lambda$. Hence, from the definition of $\mathsf{R}(\mathsf{D}, \mathsf{L})$ we get

$$\begin{aligned} \mathsf{R}(\lambda\mathsf{D}_1 + (1-\lambda)\mathsf{D}_2, \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2) &\overset{(a)}{\leq} \mathsf{R}(\mathsf{D}_\lambda, \mathsf{L}_\lambda) \\ &\leq \mathsf{I}\big(\boldsymbol{X}^{[\mathsf{M}]}; \hat{\boldsymbol{X}}_\lambda^{[\mathsf{M}]}, \boldsymbol{Q}_\lambda\big) \\ &\overset{(b)}{\leq} \lambda\mathsf{R}(\mathsf{D}_1, \mathsf{L}_1) + (1-\lambda)\mathsf{R}(\mathsf{D}_2, \mathsf{L}_2). \end{aligned}$$

where (a) holds since the minimization is taken over a smaller constrained set, and (b) follows from (B.10). This then completes the proof of convexity of $\mathsf{R}(\mathsf{D},\mathsf{L})$.

In the following, we show that for the case of memoryless vector sources $\boldsymbol{X}^{[\mathsf{M}]}$, i.e., each element of $\boldsymbol{X}^{(m)}$ is chosen independently and uniformly at random from $\mathbb{F}$, the converse bound for the achievable rate R is indeed $\mathsf{R}(\mathsf{D},\mathsf{L})$.

Consider an i.i.d. sequence of length-M vectors $\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}$, where $\boldsymbol{X}_i^{[\mathsf{M}]} = (X_i^{(1)}, X_i^{(2)}, \ldots, X_i^{(\mathsf{M})})$, $i \in [\beta]$, and each component $X_i^{(m)}$ is distributed according to $P_{X^{(m)}}$. The answer encoder f_{A}^n takes the input sequence $\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}$ and the generated query $\boldsymbol{Q}$ to construct the codewords that are indexed by $\mathcal{A} = \{1, 2, \ldots, 2^{\beta\mathsf{R}}\}$. The reconstruction decoder $f_{\hat{\mathsf{X}}}^n$ outputs an estimate $\{\hat{\boldsymbol{X}}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}$ for $\hat{\boldsymbol{X}}_i^{[\mathsf{M}]} = \big(\hat{X}_i^{(1)}, \hat{X}_i^{(2)}, \ldots, \hat{X}_i^{(\mathsf{M})}\big)$ using the answer $\boldsymbol{A}$,[2] requested index M, and the generated query $\boldsymbol{Q}$ at the user side. A feasible scheme should satisfy

$$\frac{1}{\beta}\sum_{i=1}^{\beta} \mathsf{E}_{M,\boldsymbol{Q}}\left[d(X_i^{(M)}, \hat{X}_i^{(\mathsf{M})})\right] \leq \mathsf{D} \text{ and } \rho\big(P_{\boldsymbol{Q}|M}\big) \leq \mathsf{L}.$$

Hence, using the fact that the average code length over a source code is bounded from below by the entropy of the source, e.g., see Cover & Thomas (2006, Thm. 5.4.1), we have

$$\begin{aligned}
\beta\mathsf{R} &\geq \mathsf{H}(\boldsymbol{A}|\boldsymbol{Q}) \\
&\overset{(a)}{=} \mathsf{H}(\boldsymbol{A}|\boldsymbol{Q}) - \mathsf{H}(\boldsymbol{A}|\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}, \boldsymbol{Q}) \\
&= \mathsf{I}\big(\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}; \boldsymbol{A} \,\big|\, \boldsymbol{Q}\big) \\
&\overset{(b)}{\geq} \mathsf{I}\big(\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}; \{\hat{\boldsymbol{X}}_i^{[\mathsf{M}]}\}_{i=1}^{\beta} \,\big|\, \boldsymbol{Q}\big) \\
&\overset{(c)}{\geq} \sum_{i=1}^{\beta} \mathsf{H}\big(\boldsymbol{X}_i^{[\mathsf{M}]} \,\big|\, \boldsymbol{Q}\big) - \sum_{i=1}^{\beta} \mathsf{H}\big(\boldsymbol{X}_i^{[\mathsf{M}]}, \hat{\boldsymbol{X}}_i^{[\mathsf{M}]} \,\big|\, \boldsymbol{Q}\big) \\
&= \sum_{i=1}^{\beta} \mathsf{I}\left(\boldsymbol{X}_i^{[\mathsf{M}]}; \hat{\boldsymbol{X}}_i^{[\mathsf{M}]} \,\middle|\, \boldsymbol{Q}\right), \\
&\overset{(d)}{\geq} \sum_{i=1}^{\beta} \mathsf{R}\Big(\mathsf{E}_{M,\boldsymbol{Q}}\big[d(X_i^{(M)}, \hat{X}_i^{(\mathsf{M})})\big], \rho(P_{\boldsymbol{Q}|M})\Big) \\
&\overset{(e)}{\geq} \beta \cdot \mathsf{R}\left(\frac{1}{\beta}\sum_{i=1}^{\beta} \mathsf{E}_{M,\boldsymbol{Q}}\left[d(X_i^{(M)}, \hat{X}_i^{(\mathsf{M})})\right], \rho(P_{\boldsymbol{Q}|M})\right) \\
&\geq \beta\mathsf{R}(\mathsf{D},\mathsf{L}),
\end{aligned}$$

where (a) holds since the answer $\boldsymbol{A}$ is a function of $\{\boldsymbol{X}_i^{[\mathsf{M}]}\}_{i=1}^{\beta}$ and $\boldsymbol{Q}$; (b) follows by the data processing inequality; (c) can be verified by using the chain rule of entropy and conditioning reduces entropy; (d) and (e) hold because $\mathsf{R}(\mathsf{D},\mathsf{L})$ is nonincreasing and convex.

C Soft Decision Decoding

In contrast to hard decision decoding where the server deterministically guesses exactly one $M = m$ from the query, we can also consider a soft decision decoding rule for the server. In this case, $f_{\hat{\mathsf{M}}}(\boldsymbol{Q})$ can be seen as a distribution over $[\mathsf{M}]$, i.e., $f_{\hat{\mathsf{M}}}(\boldsymbol{Q}) = F_{\hat{M}}(m|\boldsymbol{Q})$ for $m \in [\mathsf{M}]$, where $\sum_m F_{\hat{M}}(m|\boldsymbol{Q}) = 1$. Let $f_{\text{Loss}}(m, \hat{M})$ be the log-loss function defined as

$$f_{\text{Loss}}(m, \hat{M}) = \mathsf{H}(M) + \log F_{\hat{M}}(m|\boldsymbol{Q}), \tag{C.11}$$

i.e., the loss is zero when the server's guess is equally likely. Then, the expected loss function is equal to

$$\mathsf{J}(f_{\mathsf{Q}}, f_{\hat{\mathsf{M}}}) = \mathsf{H}(M) - \sum_{\boldsymbol{q}} P_{\boldsymbol{Q}}(\boldsymbol{q})\Big(-\sum_{m} P_{M|\boldsymbol{Q}}(m|\boldsymbol{q}) \log F_{\hat{M}}(m|\boldsymbol{q})\Big)$$

[2]Here, without loss of generality, we assume that the reconstruction $\hat{\boldsymbol{X}}^{[\mathsf{M}]}$ has the same dimensions as $\boldsymbol{X}^{[\mathsf{M}]}$. However, the user can choose the desired file $\hat{\boldsymbol{X}}^{(\mathsf{M})}$ to retrieve.

$$= \mathsf{H}(M) - \sum_{\boldsymbol{q}} P_{\boldsymbol{Q}}(\boldsymbol{q})\, \mathsf{H}\big(P_{M|\boldsymbol{Q}=\boldsymbol{q}}(\cdot) \;\big\|\; F_{\hat{M}}(\cdot|\boldsymbol{q})\big) \tag{C.12}$$

$$\leq \mathsf{H}(M) - \sum_{\boldsymbol{q}} P_{\boldsymbol{Q}}(\boldsymbol{q})\, \mathsf{H}\big(P_{M|\boldsymbol{Q}=\boldsymbol{q}}(\cdot)\big) = \mathsf{H}(M) - \sum_{\boldsymbol{q}} P_{\boldsymbol{Q}}(\boldsymbol{q})\, \mathsf{H}\big(M \,\big|\, \boldsymbol{Q}=\boldsymbol{q}\big), \tag{C.13}$$

where $\mathsf{H}\big(P_X \;\|\; P_Y\big)$ denotes the cross-entropy between the distributions P_X and P_Y, and (C.13) follows since for any two distributions $P_X(\cdot)$ and $P_Y(\cdot)$, we have $\mathsf{H}\big(P_X(\cdot) \;\|\; P_Y(\cdot)\big) \geq \mathsf{H}\big(P_X(\cdot)\big)$. Moreover, observe that

$$\max_{f_{\hat{\mathrm{M}}}} \big(\mathsf{J}(f_{\mathrm{Q}}, f_{\hat{\mathrm{M}}})\big) \leq \mathsf{H}\,(M) - \mathsf{H}\big(M \,\big|\, \boldsymbol{Q}\big) = \mathsf{I}\,(M\,;\boldsymbol{Q})\,,$$

and from (C.13), it follows that equality holds if $P_{M|\boldsymbol{Q}=\boldsymbol{q}}(\cdot) = F_{\hat{M}}(\cdot|\boldsymbol{q})$. Therefore, under the log-loss function, the leakage to the server is measured in terms of MI, i.e., $\rho(P_{\boldsymbol{Q}|M}) = \mathsf{I}\,(M\,;\boldsymbol{Q})$, and the user wishes to minimize the MI leakage.

D Achievable Schemes From Section 3

D.1 Compression-Based Scheme for Gaussian Data

As a reference for synthetic Gaussian data (see Fig. 3(a)), we used the compression-based scheme outlined in Section 3. As explained in Section 3, the scheme reduces to quantization of random vectors drawn from a Gaussian multinomial distribution. We used the generalized Lloyd algorithm, also known as the Linde-Buzo-Gray (or LBG) algorithm (Linde et al., 1980). Note that this algorithm is closely related to k-means clustering in unsupervised learning (Levrard, 2018). The authors argue in Linde et al. (1980) that although their approach is not guaranteed to provide the best quantizer, its results are nearly optimal for a wide class of distributions.

We briefly explain the steps of finding the quantization vectors (or levels) below and refer the interested reader to Linde et al. (1980) for further details. When quantizing a Gaussian vector into r bits, the number of quantization vectors is $\mathrm{k} = 2^{\mathrm{r}}$.

1. First, generate a large sample of n vectors $\{\boldsymbol{x}_i\}_{i=1}^{\mathrm{n}}$ from the desired multinomial Gaussian distribution.
2. Then, select an initial set of quantization vectors $\{\boldsymbol{q}_j\}_{j=1}^{\mathrm{k}}$.
3. Next, continue iterating as follows:
   (a) Each sample vector $\boldsymbol{x}_i$ is quantized to (approximated by) its closest quantization vector (its "nearest neighbor") $\boldsymbol{q}_{j_i^*}$ where
   $$j_i^* = \arg\min_j d(\boldsymbol{x}_i, \boldsymbol{q}_j).$$
   This gives a mean quantization error (or distortion) among all the sample vectors of
   $$\mathsf{D} = \frac{1}{\mathrm{n}} \sum_{i=1}^{\mathrm{n}} d(\boldsymbol{x}_i, \boldsymbol{q}_{j_i^*}).$$
   (b) Each quantization vector is updated to the mean of the sample vectors as follows:
   $$\boldsymbol{q}_j \leftarrow \frac{1}{|\mathcal{S}_j|} \sum_{\boldsymbol{x} \in \mathcal{S}_j} \boldsymbol{x},$$
   where the set of sample vectors that are quantized to $\boldsymbol{q}_j$ is denoted by $\mathcal{S}_j$ ("neighborhood" of $\boldsymbol{q}_j$).
4. The algorithm stops when the mean quantization error between two consecutive iterations changes less than a predefined threshold.

Since the mean quantization error (or distortion) D is nonnegative and nonincreasing between iterations, the algorithm is guaranteed to converge. Since it is a randomized algorithm, we ran it

multiple times and chose the quantizer that produced the smallest mean quantization error. As a final remark, we remind the reader that our goal here is not to find the best quantization possible, but only to have a good enough approximation.

Additionally, after the avhievable points are calculated, we make sure they all satisfy the convexity property (cf. Lemma 1). More precisely, if the triples $(\mathsf{R}_1, \mathsf{D}_1, \mathsf{L}_1)$, $(\mathsf{R}_2, \mathsf{D}_2, \mathsf{L}_2)$, and $(\lambda\mathsf{R}_1 + (1-\lambda)\mathsf{R}_2, \mathsf{D}_3, \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2)$ are achievable for some $0 < \lambda < 1$, we update

$$\mathsf{D}_3 \leftarrow \min(\mathsf{D}_3, \lambda\mathsf{D}_1 + (1-\lambda)\mathsf{D}_2).$$

D.2 Shannon's Scheme for Gaussian Data

This section briefly outlines the Shannon's scheme for Gaussian data (plotted in Fig. 3(a)). Following its description in Section 3, the server compresses N, $\mathsf{N} \in [\mathsf{M}]$, independent files, each consisting of β independent Gaussian RVs, together. It is known from the rate-distortion theory for multi-dimensional sources Gray (1973), as the file size $\beta \to \infty$, that a download rate $\mathsf{R} = \mathsf{N}R_{\mathrm{G}}(\mathsf{D})$ is achievable with distortion D. Here, $R_{\mathrm{G}}(\mathsf{D}) = \max\{\frac{1}{2}\log\left(\sigma^2/\mathsf{D}\right), 0\}$ for a Gaussian RV distributed according to $\mathcal{N}(\mu, \sigma^2)$ (Cover & Thomas, 2006, Ch. 10). Similar to Section D.1, we further use a convexifying approach to obtain an achievable scheme for any hard decision leakage (or accuracy) $1/\mathsf{M} \leq \mathsf{L} \leq 1$. In the following we brief describe the scheme. First, we select two accuracy values, say L_1 and L_2, $\mathsf{L}_1, \mathsf{L}_2 \in \{1, 1/2, \ldots, 1/\mathsf{M}\}$. Applying the Gaussian rate-distortion function of $R_{\mathrm{G}}(D) = \frac{1}{2}\log\left(\sigma^2/\mathsf{D}\right)$, $\mathsf{D} \in (0, \sigma^2]$, to the previously outlined scheme, one can obtain two achievable distortions $D_{\mathrm{G}}(\mathsf{R}_1, \mathsf{L}_1) = \sigma^2 \cdot 2^{-2\mathsf{R}_1\mathsf{L}_1}$ and $D_{\mathrm{G}}(\mathsf{R}_1, \mathsf{L}_1) = \sigma^2 \cdot 2^{-2\mathsf{R}_2\mathsf{L}_2}$. Next, the desired accuracy $\mathsf{L} \in [\mathsf{L}_1, \mathsf{L}_2]$ is selected to be $\mathsf{L} = \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2$, and the goal is to determine the minimum possible linear combination of $D_{\mathrm{G}}(\mathsf{R}_1, \mathsf{L}_1)$ and $D_{\mathrm{G}}(\mathsf{R}_2, \mathsf{L}_2)$ subject to $\mathsf{R} = \lambda\mathsf{R}_1 + (1-\lambda)\mathsf{R}_2$, i.e.,

$$\min_{\substack{\mathsf{L}_1^{-1}, \mathsf{L}_2^{-1} \in [\mathsf{M}] \\ \mathsf{L} = \lambda\mathsf{L}_1 + (1-\lambda)\mathsf{L}_2}} \min_{\mathsf{R} = \lambda\mathsf{R}_1 + (1-\lambda)\mathsf{R}_2} \left[\lambda D_{\mathrm{G}}(\mathsf{R}_1, \mathsf{L}_1) + (1-\lambda) D_{\mathrm{G}}(\mathsf{R}_2, \mathsf{L}_2)\right]. \tag{D.14}$$

This then gives the performance of the synthetic Gaussian dataset in Fig. 3(a). As a final remark, we also numerically evaluate the values of $\mathsf{R}(\mathsf{D}, \mathsf{L})$ for the Gaussian dataset. For instance, for $\mathsf{D} = 1.875$ and $\mathsf{L} = 0.6$, we obtain $\mathsf{R}(\mathsf{D}, \mathsf{L}) \approx 2.0345$, while (D.14) gives 1.875 for $(\mathsf{R}, \mathsf{L}) = (2.0, 0.6)$, confirming that the performance of our proposed Shannon's scheme is optimal.

D.3 Compression-Based Scheme for the MNIST Dataset

The scheme is based on the ideas of Section 3 and reduces to choosing a lossy compression method for one image. First, we applied a grayscale compression similar to the instruments used by the JPEG standard. We refer the interested reader to (Bocharova, 2010, Sec. 8.2) for more details and other techniques for image compression. The main ingredient of the JPEG standard is a two-dimensional discrete cosine transform. In the standard, an image is split into 8×8 blocks and the transform is applied to each block independently. Since the images from the MNIST dataset are of size 28×28, we applied the transform to 7×7 blocks. In this way, an image is split into 16 blocks. Next, the resulting values were quantized by a uniform scalar quantizer. For each block, the top-left coefficients play the most important role as they grasp the low-frequency contents of the block. Thus, they are assigned more bits from the available pool of bits (defined by the desired rate). Finally, these quantized coefficients were encoded with a combination of run-length encoding and a two-dimensional Huffman code.

Since the MNIST images have a small size, the entropy coding techniques do not provide any compression benefits. For example, the usually ignored overhead of storing the Huffman coding table requires too many bits in our case. Also, the run-length encoder performs poorly as there are not many repetitive values. In fact, it even increases the size of an MNIST image on many occasions. Therefore, we turned to a much simpler scalar quantization of the original grayscale images where each image is split into blocks. For each block, the average of values of its pixels was quantized and stored. This method also allows for easier control of the desired rate.

E LEARNING ALGORITHM FOR THE DATA-DRIVEN APPROACH

Algorithm 1: Training algorithm for generative adversarial user privacy

Input : Number of training samples $\mathtt{n}$, training samples $(\boldsymbol{X}_1^{[\mathsf{M}]}, \ldots, \boldsymbol{X}_{\mathtt{n}}^{[\mathsf{M}]})$, number of training iterations T, size of minibatch for stochastic gradient decent $\mathtt{b}$, and initial tuning parameter η_{initial}

Output : f_{Q}, f_{A}, $f_{\hat{\mathrm{X}}}$, and $f_{\hat{\mathrm{M}}}$

$t \leftarrow 1$
$\eta \leftarrow \eta_{\text{initial}}$
Initialize the neural networks representing f_{Q}, f_{A}, $f_{\hat{\mathrm{X}}}$, and $f_{\hat{\mathrm{M}}}$
while $t \leq \mathsf{T}$ **do**
  for $m \in [\mathsf{M}]$ **do**
    Generate a minibatch $\mathcal{B}_m \subseteq [\mathtt{n}]$ of indices corresponding to the m-th file, $|\mathcal{B}_m| = \mathtt{b}$
    Generate $\mathtt{b}$ noise samples $\boldsymbol{s}_{\mathcal{B}_m}$ from the distribution $\mathcal{N}(\mathbf{0}, I)$
    for $b \in \mathcal{B}_m$ **do**
      $\boldsymbol{q}_{m,b} \leftarrow f_{\mathrm{Q}}(m, \boldsymbol{s}_b)$
    end
  end
  Update $f_{\hat{\mathrm{M}}}$ by descending its stochastic gradient:
  $\sum_{m=1}^{\mathsf{M}} \sum_{b \in \mathcal{B}_m} \frac{1}{\mathtt{b} \times \mathsf{M}} \log F_m(\boldsymbol{q}_{m,b})$
  for $m \in [\mathsf{M}]$ **do**
    Generate a minibatch $\mathcal{B}_m \subseteq [\mathtt{n}]$ of indices corresponding to the m-th file, $|\mathcal{B}_m| = \mathtt{b}$
    Generate $\mathtt{b}$ noise samples $\boldsymbol{s}_{\mathcal{B}_m}$ from the distribution $\mathcal{N}(\mathbf{0}, I)$
    Generate a minibatch of $\mathtt{b}$ training samples $\{\boldsymbol{X}_b^{(m)} : b \in \mathcal{B}_m\}$
    for $b \in \mathcal{B}_m$ **do**
      $\boldsymbol{q}_{m,b} \leftarrow f_{\mathrm{Q}}(m, \boldsymbol{s}_b)$
    end
  end
  for $m \in [\mathsf{M}]$ **do**
    for $b \in \mathcal{B}_m$ **do**
      $\boldsymbol{a}_{m,b} \leftarrow f_{\mathrm{A}}(\boldsymbol{q}_{m,b}, \boldsymbol{X}_b^{([\mathsf{M}])})$
    end
  end
  Update f_{Q}, f_{A}, and $f_{\hat{\mathrm{X}}}$ by descending its stochastic gradient:
  $\sum_{m=1}^{\mathsf{M}} \sum_{b \in \mathcal{B}_m} \frac{1}{\mathtt{b} \times \mathsf{M}} \left[\log F_m(\boldsymbol{q}_{m,b}) + \eta \cdot d\left(\boldsymbol{X}_b^{(m)}, f_{\hat{\mathrm{X}}}(\boldsymbol{a}_{m,b}, m, \boldsymbol{q}_{m,b})\right)\right]$
  Update the tuning parameter η
  $t \leftarrow t + 1$
end
return f_{Q}, f_{A}, $f_{\hat{\mathrm{X}}}$, *and* $f_{\hat{\mathrm{M}}}$

As elaborated in Section 4.1, learning is done by first fixing the download rate R and then solving the minimax optimization problem in (9). Here, the rate is computed based on the dimension of the answer network's output and the corresponding quantizer levels like Blau & Michaeli (2019); Mentzer et al. (2018). In Algorithm 1 we outline the detailed training algorithm, which is similar to Algorithm 1 in Huang et al. (2017). The learning rate is selected empirically for each accuracy and distortion level with the RMSprop optimizer, and the number of iterations is $\mathsf{T} = 100000$. The number of training samples, denoted by $\mathtt{n}$, for the Gaussian case is set to a sufficiently large number (samples can easily be generated from a Gaussian RV) and to 60000 (6000 for each digit) for the MNIST dataset.

The architectures of the deep neural networks representing the functions f_{Q}, f_{A}, $f_{\hat{\mathrm{X}}}$, and $f_{\hat{\mathrm{M}}}$ used for training for the synthetic Gaussian dataset are detailed in Table 1. In the special case of accuracy $\mathsf{L} = 1$, i.e., full leakage, the problem reduces to the classical rate-distortion problem, and to improve performance, we only considered an encoder (for compression of the requested file), corresponding to f_{A}, and a decoder (for decompression at the user side), corresponding to $f_{\hat{\mathrm{X}}}$. Both the encoder

and the decoder are represented as deep neural networks and trained in the classical way (Kingma & Welling, 2014).

Table 1: Neural network architectures used for training for the synthetic Gaussian dataset. FC stands for fully connected layer. Size is the number of neurons or input size. As activation function we used the scaled exponential linear unit (SeLU) (Klambauer et al., 2017).

Query generator, f_{Q}

Size	Layer
8	Input
8	FC, SeLU
8	FC, SeLU
8	FC, SeLU
8	FC, SeLU
4	FC, SeLU

Answer generation, f_{A}

Size	Layer
$4 + 3 \times 4 \times 1$	Input
16	Flatten
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
Answer dim	FC, tanh

Decoder, $f_{\hat{\mathrm{X}}}$

Size	Layer
Answer dim $+8$	Input
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
256	FC, SeLU
3	FC

Adversary, $f_{\hat{\mathrm{M}}}$

Size	Layer
4	Input
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
4	FC, softmax

For the MNIST dataset there are $\mathsf{M} = 10$ files (there are 10 digits) and each file is of size $\beta = 28 \times 28 = 784$ symbols, or $784 \times 8 = 6272$ bits (each picture is of size 28×28 pixels and each pixel is of size 8 bits). The architectures of the deep neural networks representing the functions f_{Q}, f_{A}, $f_{\hat{\mathrm{X}}}$, and $f_{\hat{\mathrm{M}}}$ in this case are given in Table 2. As for the Gaussian case, training for MNIST can be simplified for $\mathsf{L} = 1$ as described above.

In order to assess the quality of the learning, we tabulate below an estimate of the inference accuracy of a MAP adversary for M, denoted by $\mathsf{L}_{\mathsf{MAP}}$, operating directly on the queries from the query network and also an estimate of the entropy of the answer $\boldsymbol{A}$ from the answer network for the Gaussian dataset for different levels of per-symbol squared error distortion D. The estimates have been found by Monte-Carlo simulation of the networks, i.e., by repeatedly generating queries and answers and estimating the probability density functions of the random variables $\boldsymbol{Q}|M$ and $\boldsymbol{A}$. For comparison we also show the inference accuracy L of the adversary network in the third line. As can be seen by comparing the second and third lines, the inference accuracy of the adversary network is close to the inference accuracy of a MAP adversary. Likewise, the entropy $\mathsf{H}(\boldsymbol{A})$ is close of $\mathsf{R} \cdot \beta$, which gives an indication that the performance of neural networks adversary is close to an optimal bound.

Table 2: Neural network architectures used for training for the MNIST dataset. FC stands for a fully connected layer. Conv and ConvT stand for a convolutional and a transposed convolutional layer, respectively, where "st" is shorthand for stride. Size is the number of neurons or input size. As activation function we used SeLU and hyperbolic tangent (tanh).

Query generator, f_{Q}

Size	Layer
20	Input
20	FC, SeLU
20	FC, SeLU
20	FC, SeLU
10	FC, SeLU

Answer generation, f_{A}

Size	Layer
$10 + 10 \times 28 \times 28$	Input
$26 \times 26 \times 8$	Conv, SeLU
$12 \times 12 \times 8$	Conv (st $= 2$), SeLU
$10 \times 10 \times 16$	Conv, SeLU
$4 \times 4 \times 16$	Conv (st $= 2$), SeLU
256	Flatten
2560	Concatenate 10 parallel outputs of previous layers
1024	FC, SeLU
1024	FC, SeLU
1024	FC, SeLU
1024	FC, SeLU
512	FC, SeLU
512	FC, SeLU
512	FC, SeLU
512	FC, SeLU
392	FC , tanh

Decoder, $f_{\hat{\mathrm{X}}}$

Size	Layer
412	Input
512	FC, SeLU
512	FC, SeLU
512	FC, SeLU
512	FC, SeLU
$4 \times 4 \times 32$	Unflatten
$11 \times 11 \times 64$	ConvT (st $= 2$), SeLU
$25 \times 25 \times 128$	ConvT (st $= 2$), SeLU
$28 \times 28 \times 1$	ConvT (st $= 1$), tanh

Adversary, $f_{\hat{\mathrm{M}}}$

Size	Layer
10	Input
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
10	FC, softmax

Gaussian data set and $\mathsf{R} = 2$

D	6.60	5.46	4.35	1.16
$\mathsf{L}_{\mathsf{MAP}}$	0.26	0.38	0.50	1.0
L	0.25	0.37	0.50	1.0
$\mathsf{H}(\boldsymbol{A})$	5.83	5.85	5.90	5.89

Gaussian data set and $\mathsf{R} = 4$

D	3.93	1.15	0.72	0.21
$\mathsf{L}_{\mathsf{MAP}}$	0.27	0.53	0.76	1.0
L	0.26	0.51	0.76	1.0
$\mathsf{H}(\boldsymbol{A})$	11.82	11.56	11.45	10.86

Table 3: Neural network architectures used for training for the Cifar-10 dataset. FC stands for a fully connected layer. Conv and ConvT stand for a convolutional and a transposed convolutional layer, respectively, where "st" is shorthand for stride. Size is the number of neurons or input size. As activation function we used SeLU, sigmoid, and hyperbolic tangent (tanh).

Query generator, f_{Q}

Size	Layer
20	Input
20	FC, SeLU
20	FC, SeLU
9	FC, SeLU
7	FC, SeLU
5	FC, SeLU

Answer generation part 1, f_{A}

Size	Layer
5	Input
5	FC, SeLU
7	FC, SeLU
9	FC, SeLU
10	FC, softmax

Answer generation part 2, f_{A}

Size	Layer
$10 \times 32 \times 32 \times 3$	Input
$10 \times 30 \times 30 \times 8$	Conv, SeLU
$10 \times 14 \times 14 \times 8$	Conv (st $= 2$), SeLU
$10 \times 12 \times 12 \times 16$	Conv, SeLU
2304×10	Concatenate 10 parallel outputs of previous layers

Answer generation part 3, f_{A}

Size	Layer
$10 + 2304 \times 10$	Input
2304×10	Broadcast and elementwise multiply
23040	Flatten
5	Input
2304	FC, SeLU
1024	FC, SeLU
Answer dim	FC, sigmoid

Decoder, $f_{\hat{\text{X}}}$

Size	Layer
Answer dim + 20	Input
512	FC, SeLU
512	FC, SeLU
512	FC, SeLU
$4 \times 4 \times 32$	Unflatten
$8 \times 8 \times 16$	ConvT (st $= 2$), SeLU
$16 \times 16 \times 8$	ConvT (st $= 2$), SeLU
$32 \times 32 \times 3$	ConvT (st $= 2$), tanh

Adversary, $f_{\hat{\text{M}}}$

Size	Layer
5	Input
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
64	FC, SeLU
10	FC, softmax