Proactive Privacy-preserving Learning for Retrieval

Peng-Fei Zhang,[1] **Zi Huang,** [1] **Xin-Shun Xu** [2]
[1] School of Information Technology & Electrical Engineering, University of Queensland, Brisbane, Australia
[2] School of Software, Shandong University, Jinan, China
mima.zpf@gmail.com, huang@itee.uq.edu.au, xuxinshun@sdu.edu.cn

Abstract

Deep Neural Networks (DNNs) have recently achieved remarkable performance in image retrieval, yet posing great threats to data privacy. On the one hand, one may misuse a deployed DNNs based system to look up data without consent. On the other hand, organizations or individuals would legally or illegally collect data to train high-performance models outside the scope of legitimate purposes. Unfortunately, less effort has been made to safeguard data privacy against malicious uses of DNNs. In this paper, we propose a data-centric Proactive Privacy-preserving Learning (PPL) algorithm for hashing based retrieval, which achieves the protection purpose by employing a generator to transfer the original data into the adversarial data with quasi-imperceptible perturbations before releasing them. When the data source is infiltrated, the adversarial data can confuse menacing retrieval models to make erroneous predictions. Given that the prior knowledge of malicious models is not available, a surrogate retrieval model is instead introduced acting as a fooling target. The framework is trained by a two-player game conducted between the generator and the surrogate model. More specifically, the generator is updated to enlarge the gap between the adversarial data and the original data, aiming to lower the search accuracy of the surrogate model. On the contrary, the surrogate model is trained with the opposing objective that is to maintain the search performance. As a result, an effective and robust adversarial generator is encouraged. Furthermore, to facilitate an effective optimization, a Gradient Reversal Layer (GRL) module is inserted to connect two models, enabling the two-player game in a one-step learning. Extensive experiments on three widely-used realistic datasets prove the effectiveness of the proposed method.

Introduction

In recent decades, Deep Neural Networks (DNNs) have made a giant leap forward and been widely adopted in image retrieval (Babenko et al. 2014; Li et al. 2017; Shen et al. 2018; Wang et al. 2020). With sufficient training data, one may build deep models with promising performance on target searching tasks. On the flip side, once misused, this advanced technology could raise significant data privacy issues. Nowadays, unprecedented amounts of user-generated images are shared or published on the Internet, which may contain sensitive information, such as faces, child pictures, ID photos, etc. A malicious search engine could use a snapshot of someone as a query example to search against the entire Internet. Based on the returned search results (i.e., relevant or similar images), implicit connections between queries and results could be exposed and revealed, causing privacy leakage. For instance, a profile photo could be the hint to find out his/her family photos. A malicious party can also crawl and scrape images online and accordingly build high-performance retrieval models used for profit or a nefarious purpose. These worrying issues highlight the necessity of protecting data privacy.

Recently, a wide range of studies have revealed the fragile nature of deep models to the existence of adversarial examples (Biggio et al. 2013; Mopuri, Garg, and Babu 2017; Gu, Dolan-Gavitt, and Garg 2017; Shafahi et al. 2018). By slightly modifying clean data, one can craft visually plausible adversarial data but mislead target models towards wrong predictive results with high probability. Such an intriguing property provides inspiration for privacy protection against malevolent ends, i.e., using adversarial data. In real world situations, it is very common that data owners are usually unaware of when and how an invasion takes place. Based on this observation, it is natural to proactively take action on the raw data before releasing them for a precaution, reducing the chance of being invaded.

Motivated by this, we propose PPL, a data-centric Proactive Privacy-preserving Learning algorithm against potential menacing models. We discuss our work in the context of hashing based searching scenarios (Shen et al. 2009; Zhu et al. 2013; Li, Wang, and Kang 2016; Yang et al. 2016; Liu et al. 2019), where high dimensional data would be represented as compact binary codes via hash techniques to satisfy the requirements of both query speed and accuracy. The proposed PPL achieves the privacy protection by transferring data from the original domain into the adversarial domain via an adversarial generator without significantly sacrificing the visual quality. As illustrated in Figure 1, before the data release, the adversarial processing is performed, where the raw data is imperceptibly modified. The perturbed data, i.e., adversarial data, will manipulate the predictive behaviours of malicious retrieval models in various scenarios. More specifically, in Figure 1 (a), when someone leverages well-established models to retrieve the released adversarial

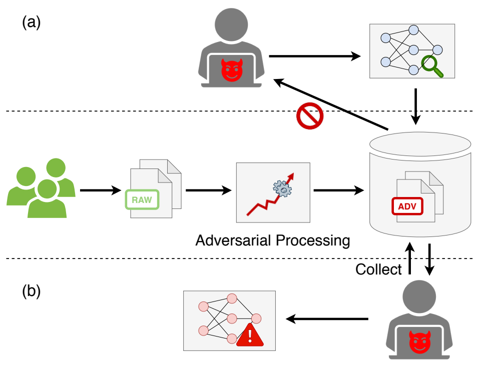


Figure 1: Data-centric proactive privacy preserving. The protection mechanism is implemented before the data release, when the raw data is transferred into the adversarial data with imperceptible adjustments. When the data source is penetrated, the modified can successfully fail the malicious users in both (a) searching with existing models (b) constructing new models.

data or take these data as queries to perform searching, their target data items would not be returned. At the same time, malicious parties who have successfully harvested the adversarial data can hardly build up capable models in retrieval tasks, where clean data is involved no matter as queries or searching targets. As stated before, malicious models are usually hidden from victims, so that we build a surrogate retrieval model as an imaginary fooling target of the generator. It is feasible as indicated by (Papernot, McDaniel, and Goodfellow 2016; Tramèr et al. 2017; Ilyas et al. 2019) that data crafted to mislead one model also interferes with other models trained for the same task.

The proposed framework is trained by a two-player game conducted between the generator and the surrogate model, where two models are optimized in the opposite direction. In detail, the generator is trained to minimize a privacy-preserving loss to enlarge the domain gap, distort the similarity relationships and corrupt the matching between adversarial data and clean data, aiming to lower the prediction accuracy of the surrogate model in the above complex privacy-preserving scenarios. In the meanwhile, the surrogate model strives to mitigate the adverse influence of the generator and maintain the performance. The adversarial game impels the generator to produce effective and robust adversarial data. To facilitate an effective optimization, a GRL module is inserted between two models, enabling the two-player game in a one step learning. Extensive experiments demonstrate the superiority of the proposed method. The main contributions are summarized as follows.

- To the best of our knowledge, this is the first work that considers privacy protection in large-scale image retrieval. The proposed PPL is developed to transfer clean data into adversarial data with imperceptible adjustment before the data release, targeting to manipulate the predictive behaviours of potential malicious retrieval models.
- A novel framework with an effective objective function is designed, where an adversarial game is played between the adversarial generator and the surrogate retrieval model with various constraints imposed. A GRL module is additionally applied to facilitate the training.
- Extensive experiments on three widely-used benchmark datasets verify the feasibility of the proposed method, including high privacy-preserving success rate and remarkable transferability.

Related work

In this section, we briefly review the most related subjects of the studied problem, including adversarial examples and deep hashing.

Adversarial Examples

(Szegedy et al. 2013) first make an assumption that there exist adversarial examples with imperceivable perturbations from original clean data that can fool deep models to make wrong predictions, and verify it by an optimization-based method. To accelerate the generation of adversarial data, (Goodfellow, Shlens, and Szegedy 2014) focus on the linear nature of neural networks and hereby design the Fast Gradient Sign Method (FGSM) to obtain an optimal max-norm constrained perturbation. (Moosavi-Dezfooli, Fawzi, and Frossard 2016) study the adversarial instability and accordingly produce adversarial examples with minimal perturbations that sufficiently change the output of the classifier. Considering training data is not always accessible, (Mopuri, Garg, and Babu 2017) misfire features at each individual layer to make target networks produce erroneous predictions. To boost the transferability of adversarial examples, (Dong et al. 2018) present a momentum-based iterative scheme, which significantly strengthens the fooling rate. Despite the promising results, these methods serve simple attacking tasks and usually act on a small proportion of data, incapable of matching the privacy-protecting purpose.

Deep Hashing

Deep hashing techniques have been widely applied in large-scale retrieval tasks in virtue of their promising performance and low computational cost. Some representative methods include Deep Supervised Hashing (DSH) (Liu et al. 2016), Deep Supervised Discrete Hashing (DSDH) (Li et al. 2017), Graph Convolutional Network Hashing (GCNH) (Zhou et al. 2020), ADSH (Jiang and Li 2018) and Semantic-Aware DIscrete Hashing (SADIH) (Zhang et al. 2019). More concretely, DSH first pairs image inputs with semantic labels as the relationship indicator, and outputs hashing codes near the discrete values in a Hamming space. To pursue reliable binary codes, DSDH proposes a discrete learning framework, where both similarity information and label information are utilized to provide sufficient supervised information. In virtue of powerful Graph Convolutional Networks (GCNs), GCNH further exploits the intrinsic semantic structure of data in order to learn more discriminative binary codes. To avoid the time-consuming training, ADSH designs an asymmetric deep hashing learning strategy, which directly learns binary codes for the entire database and produces hash functions for query points. SADIH is a discrete asymmetric similarity-preserving learning framework, where a latent semantic space is constructed in order to learn discriminative embeddings.

Method

Preliminaries

Let $X = \{x_i\}_{i=1}^{n}$ denote a dataset with n clean samples. $S \in \{-1,1\}^{n\times n}$ is the semantic similarity matrix, where $S_{ij} = 1$ if the i-th and the j-th data points are semantically similar and $S_{ij} = -1$ otherwise. The goal of hashing is to learn a hash projection function $\mathcal{F}(\cdot;\vartheta_f) : X \to H \in [-1,1]^{c\times n}$, where c is the code length and $\mathcal{F}(\cdot;\vartheta_f)$ is a deep network with the parameter ϑ_f in deep hash learning. The final binary codes can be obtained by applying a $sign$ function to H:

$$B = sign\,(H) \in \{-1,1\}^{c\times n}. \tag{1}$$

The ultimate goal of our study is to learn an adversarial generator to transfer the clean data $x_i \in X$ into the adversarial data $z_i \in Z$ with the normal use guarantee, which can mislead the test-time behaviours of retrieval models.

Overview

The architecture of the proposed method is illustrated in Figure 2, which consists of an adversarial generator $\mathcal{G}$, a Gradient Reversal Layer (GRL) module Re and a surrogate retrieval model $\mathcal{F}$. The generator is tasked to craft corresponding adversarial examples for clean images, while the retrieval model is employed to learn binary representations for input images. The GRL module is inserted between both to enable the training to proceed in a one step way by minimizing the affinity-preserving loss $\mathcal{J}_o$ and the privacy-preserving loss $\mathcal{J}_p$.

Adversarial Examples

Here, we adopt a data-dependent adversarial data generation strategy using an auto-encoder-like generator. Detailedly, given a normal image x_i, the generator $\mathcal{G}$ takes it as input and accordingly outputs a noise pattern, which is then injected into the original image to craft the adversarial counterpart z_i. Principally, the generated adversarial image should be perceptually indistinguishable from its original version. To this end, we impose a $l_p, p \in \{1,2,\infty\}$ constraint. The generation process is formulated as follows:

$$\begin{aligned} &z_i = x_i + \mathcal{G}(x_i;\vartheta_g), \\ &s.t.\ \|\mathcal{G}(x_i;\vartheta_g)\|_p \le \epsilon, p \in \{1,2,\infty\}, \end{aligned} \tag{2}$$

where ϑ_g is the parameter of the generator $\mathcal{G}$ and ϵ is the magnitude of the perturbation.

Binary Representation Learning

Through the hierarchical processing of the surrogate retrieval model $\mathcal{F}$, the binary representations for x_i and z_i can be obtained as follows:

$$\begin{aligned} h(x_i) &= sign\,(\mathcal{F}(x_i;\vartheta_f)) \in \{-1,1\}^{c\times 1}, \\ h(z_i) &= sign\,(\mathcal{F}(z_i;\vartheta_f)) \in \{-1,1\}^{c\times 1}, \end{aligned} \tag{3}$$

where ϑ_f is the parameter of $\mathcal{F}$ and c is the code length.

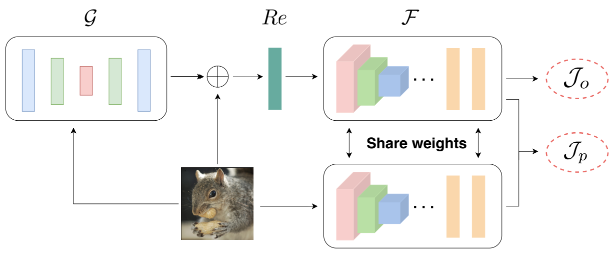


Figure 2: Illustration of the proposed method.

Affinity-preserving Loss

We wish the adversarial shift would not influence the normal uses of data. To achieve this, in Eq. (2), we have made a constraint that the perturbations would not significantly affect the image quality. Besides, an affinity-preserving loss is introduced to restrict that the generated adversarial data preserves well the original similarity structure in the representation space, such that they can be used for proper purposes. For example, data owners may share data with researchers who then utilize these data to build models to test new ideas. Formally, the loss is defined as follows:

$$\mathcal{J}_o = \sum_{i,j=1}^{n} \|h(z_i)^T h(z_i) - c \cdot S_{ij}\|_F^2. \tag{4}$$

Privacy-preserving Loss

The application of adversarial data would induce four types of searching scenarios: (1) query points and database points are all clean data, (2) queries are clean while database data is adversarial, (3) queries are adversarial and database points are clean, (4) query points and database points are all adversarial. To protect privacy, we wish any model trained on one type of data can not work when facing the other type of data. For example, a model trained on clean data can only make correct predictions within clean domain, i.e., the case (1), while failing in performing searching between clean data and adversarial data or within adversarial domain, i.e., the cases (2), (3) and (4). To enable this, we introduce a privacy-preserving loss that consists of a domain loss, a similarity loss and a matching loss, to manage the difference between clean data and adversarial data.

Domain Loss The domain loss is tasked to control the domain discrepancy between clean data and adversarial data, thereby deciding if a model trained on one domain (e.g., adversarial domain) can work on the other domain (e.g., clean domain). Inspired by the Contrastive Domain Discrepancy (CDD) measure in (Kang et al. 2019), we propose the Similarity-aware Maximum Mean Discrepancies (SMMD) to measure the domain disparity between clean data and adversarial data with similarity relationship considered, i.e., the intra-similarity discrepancy and the inter-similarity discrepancy. More specifically, denote the conditional distributions for clean domain X and adversarial domain Z as $P(h(X)|S)$ and $Q(h(Z)|S)$, respectively. SMMD conducts a two-sample statistical test and accordingly decides whether or not to accept the null hypothesis $P = Q$. When the difference of statistics between two domains is larger, the two distributions are considered to differ from each other with higher probability, or vice versa. SMMD calculates the difference between the mean embeddings of two distributions P and Q in the reproducing kernel Hilbert space (RKHS) with the squared formulation defined as follows:

$$\mathcal{D}_s \triangleq \sup_{f\sim\mathcal{H}} \left\|\mathbb{E}_P\left[f(h(X)|S)\right] - \mathbb{E}_Q\left[f(h(Z)|S)\right]\right\|_{\mathcal{H}}^2, \quad (5)$$

where $\mathcal{H}$ is a class of functions, $\mathbb{E}$ is the expectation with respect to the given distribution defined as follows:

$$\begin{aligned} &\mathbb{E}_{x\sim p}\left[f\left(x\right)\right] = \left\langle f\left(x\right), \mu\right\rangle_{\mathcal{H}}, \\ &s.t.\ \forall f \sim \mathcal{H}, \end{aligned} \quad (6)$$

where μ is the mean embedding of the distribution p. According to the two-sample statistical test theory (Gretton et al. 2012), given rich $\mathcal{H}$, the $\mathcal{D}_s$ satisfies that if and only if $\mathcal{D}_s = 0$, $P = Q$.

Denoting $S_{ij}^{+} = \begin{cases} 1 & S_{ij} > 0 \\ 0 & S_{ij} \le 0 \end{cases}$, $S_{ij}^{-} = \begin{cases} 1 & S_{ij} \le 0 \\ 0 & S_{ij} > 0 \end{cases}$, the unbiased estimation for $\mathcal{D}_s$ is defined as follows:

$$\widehat{\mathcal{D}}_s = \widehat{\mathcal{D}}_s^{+} - \widehat{\mathcal{D}}_s^{-}, \quad (7)$$

where $\widehat{\mathcal{D}}_s^{+}$ measures the gap between similar data across domains while $\widehat{\mathcal{D}}_s^{-}$ calculates the discrepancy between dissimilar data, both of which are defined as follows:

$$\begin{aligned} \widehat{\mathcal{D}}_s^{*} = &\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{S_{ij}^{*}\cdot k(h(x_i), h(x_j))}{\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{*}} \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{S_{ij}^{*}\cdot k(h(z_i), h(z_j))}{\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{*}} \\ &- 2\cdot\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{S_{ij}^{*}\cdot k(h(x_i), h(z_j))}{\sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij}^{*}}, \\ s.t.\ &* \in \{+,-\}, \end{aligned} \quad (8)$$

where k is the kernel selected for the mapping function h.

Different from the original Maximum Mean Discrepancies (MMD) that treats each sample equally, SMMD further takes into consideration the intrinsic similarity structure when measuring the discrepancy between two domains. In light of this, SMMD can reliably discriminate whether samples from different domains should be aligned or not. For instance, minimizing SMMD means narrowing the gap between similar samples from two domains, while enlarging the discrepancy between data that is dissimilar. In comparison, MMD can be minimized even when dissimilar samples from different domains are misaligned.

Similarity Loss Another case that needs to be considered in privacy-preserving learning is when query points and database points are from heterogeneous sources, i.e., one is adversarial data and the other is clean data. To cope with it, the pair-wise similarity loss between clean data and adversarial data is introduced to decide if searching between two domains are available:

$$\mathcal{J}_s = \sum_{i,j=1}^{n} \left\|(h(x_i))^T h(z_i) - c\cdot S_{ij}\right\|_F^2. \quad (9)$$

The above similarity loss measures whether the adversarial perturbations change the original similarity structure. When the loss is minimized, the similarity between two kinds of data is maintained, which means the search between data from two domains is enabled, or vice versa.

Matching Loss To further control the difference between clean data and its adversarial counterpart, we impose the matching loss between each clean-adversarial pair in the final representation space:

$$\mathcal{J}_e = \sum_{i=1}^{n} \left\|h(x_i) - h(z_i)\right\|_F^2. \quad (10)$$

Incorporating Eq. (7), (9) and (10) together, we can get the final privacy-preserving loss that measures the gap between two sets of data:

$$\mathcal{J}_p = \alpha\widehat{\mathcal{D}}_s + \beta\mathcal{J}_s + \gamma\mathcal{J}_e, \quad (11)$$

where α, β, γ are balance parameters.

Objective Function

To achieve the privacy protection, we seek the optimal parameter of the generator that maximizes the domain gap, distort the similarity relationships and corrupt the matching between adversarial data and clean data by maximizing the loss $\mathcal{J}_p$. Furthermore, we train the retrieval model by minimizing the loss to neglect the adverse influence of the adversarial perturbations, thereby encouraging the generator to produce effective and robust adversarial data. In addition, the affinity-preserving loss $\mathcal{J}_o$ needs to be minimized. To this end, the optimization problem is to find the parameters $\hat{\vartheta}_g$ and $\hat{\vartheta}_f$ that jointly satisfy:

$$\begin{aligned} \hat{\vartheta}_g &= \arg\min_{\vartheta_g} \mathcal{J}_o(\vartheta_g, \vartheta_f) - \mathcal{J}_p(\vartheta_g, \vartheta_f), \\ \hat{\vartheta}_f &= \arg\min_{\vartheta_f} \mathcal{J}_o(\vartheta_g, \vartheta_f) + \mathcal{J}_p(\vartheta_g, \vartheta_f). \end{aligned} \quad (12)$$

Optimization

Training the whole network requires the backpropagation with the objective (12). Nevertheless, there are two major difficulties to solve the optimization problem. One problem is the discrete constraint imposed on the binary code learning as shown in Eq. (3), where the $sign(\cdot)$ would lead to an intractable back-propagate gradient problem. To deal with it, we use the $tanh(\cdot)$ function to approximate the $sign(\cdot)$ function. The other obstacle lies in the two-player game, where the opposing objectives in privacy preserving make it hard to update the whole model in a one-step training manner. To approach the challenging issue, we insert the Gradient Reversal Layer (GRL) (Ganin and Lempitsky 2015) between two modules. In particular, GRL performs an identity transform during the forward propagation, while the gradient from the retrieval model would be multiplied by a negative gradient reversal factor $-\lambda$ during the backpropagation. Specifically, let $Re(\cdot)$ denote the GRL. The forward

Algorithm 1 Proactive Privacy-preserving Learning

```
Input: Training data: $X$, mini-batch size: $m$, learning rate: $\sigma_g, \sigma_f$, hash code length: $c$, iteration times: $t$, epoch times: $e = [t/m]$
Output: Adversarial generator: $\mathcal{G}(\cdot; \vartheta_g)$
Procedure:
Randomly initialize $\vartheta_g, \vartheta_f$
for $i = 1 : t$ do
    for $j = 1 : e$ do
        1. $x_i \sim X$ // Construct a mini-batch
        2. $\vartheta_g \leftarrow \vartheta_g - \sigma_g \nabla_{\theta_g}(\mathcal{J}_o - \mathcal{J}_p)$ // Update $\vartheta_g$ by SGD
        3. $\vartheta_f \leftarrow \vartheta_f - \sigma_f \nabla_{\theta_f} \mathcal{J}_p$ // Update $\vartheta_f$ by SGD
    end for
end for
Return: $\vartheta_g$
```

and backpropagation behaviours can be formalized as follows:

$$\begin{aligned} &Re(x) = x, \\ &\frac{dRe}{dx} = -\lambda \cdot I, \end{aligned} \tag{13}$$

where I is an identity matrix, x is any input. In our framework, the gradient reversal is applied in the backpropagation with the loss $\mathcal{J}_p$ while the other is maintained unchanged. As a result, the objective becomes:

$$\begin{aligned} &(\hat{\vartheta}_g, \hat{\vartheta}_f) = \arg\min_{\vartheta_g, \vartheta_f} \mathcal{J}(\vartheta_g, \vartheta_f), \\ s.t. \quad &\mathcal{J}(\vartheta_g, \vartheta_f) = \mathcal{J}_o(\vartheta_g, \vartheta_f) + \mathcal{J}_p(\vartheta_g, \vartheta_f). \end{aligned} \tag{14}$$

By the discrete relaxation and the gradient reversal operation, we can achieve a one-step training behaviour through the stochastic gradient descent (SGD). The optimization scheme is summarized in **Algorithm 1**, where the generator and the surrogate model are updated iteratively.

Experiments

Datasets To validate the effectiveness of the proposed PPL, we conduct extensive experiments on three widely-used benchmark datasets, i.e., CIFAR-10 (Krizhevsky et al. 2009), ImageNet (Deng et al. 2009), and FashionMNIST (Xiao, Rasul, and Vollgraf 2017). In detail, CIFAR-10 contains 60,000 images, each of which is annotated with one of 10 semantic class labels. Thereinto, 59,000 images are sampled from the dataset to constitute the retrieval set and the remaining 1000 samples are used for testing. ImageNet is an important benchmark dataset for computer vision related tasks, which consists of more than 1.2M images. Each image is labeled with one of the 1,000 categories. We select a subset of 13 categories amounts to 16,900 samples to constitute the retrieval database, and 1,300 images for testing. FashionMNIST is an article image dataset-consisting of a test set of 10,000 grayscale examples and a database of 60,000 grayscale examples. Each sample belongs to a label from 10 classes. In our experiment, for there datasets, 10,000 data points are randomly selected from the retrieval database as the training set.

Baselines and Evaluation Metric

To test the validity of the propose PPL, two representative deep hash learning methods, i.e., ADSH (Jiang and Li 2018) and DSDH (Li et al. 2017), are selected as the targeted fooling retrieval systems. The widely-used Mean Average Precision (mAP) which can well reflect both ranking information and precision is selected as the evaluation criterion.

Implementation Details

For experiments on CIFAR-10 and ImageNet, ResNet50 (He et al. 2016) network is adopted as the backbone of the surrogate retrieval model. In details, we remove the last layer of ResNet50 and add a fully connect (fc) layer which contains c hidden units on top of remaining layers. As images in FashionMNIST are grayscale, we construct a simple deep network composed of 2 convolution layers with respective 20 and 50 channels, on the top of which is a fc layer with c hidden units. Except for the last layer where the $tanh(\cdot)$ activation function is adopted, the $ReLU(\cdot)$ activation function is applied in the rest of layers. For fairness, two test models are based on the same backbone. With respect to the adversarial generator, we adopt an auto-encoder like structure, where the encoder/decoder consists of 4 convolution layers with channel numbers set as 64,128,256,512 for CIFAR-10 and ImageNet, and 16,32,64,128 for FashionMNIST.

The parameters are set as follows. we set $\alpha = \gamma = \lambda = 1, \beta = 10$. The perturbation constraints are selected as the l_∞ norm with ϵ as 0.1,0.032,0.3 for CIFAR-10, ImageNet and FashionMNIST, respectively. The generated images would be projected into $[0, 1]$ space. The learning rates are 10^{-4} and 10^{-3} for the generator and the retrieval model. The iterations are respectively fixed as 500,500 and 1000. We fix the batch sizes as 32,32,512. The experiments are implemented with Pytorch on a workstation (with Intel XEON E5-2650 v3 @ 2.60GHz CPU, NVIDIA 1080Ti GPU).

Results and Discussions

Following the **Algorithm 1**, we train the model and then leverage the well-trained generator to produce adversarial data to test the privacy-preserving performance. Due to space limitations, we only report test results when using the generator learned from PPL with 8-bit code length. There are four searching scenarios included: "$C \to C$" (using clean data to query clean data), "$C \to A$" (using clean data to query adversarial data), "$A \to C$" (using adversarial data to query clean data), "$A \to A$" (using adversarial data to query adversarial data).

mAP Results The mAP results on different datasets are summarized in Table 1. Thereinto, "C" and "A" in the column "Train" represent that target models are trained on clean data and adversarial data, respectively. From the results, we can have the following observations.

On the one hand, when trained on clean data, both ADSH and DSDH achieve promising results in the normal search task, i.e., both query points and database data are clean. However, when facing adversarial data in the query set or the retrieval database, the performance of both methods drops dramatically, clearly evidencing the efficacy of the

Train	Method	Task	CIFAR-10				ImageNet				FashionMNIST			
			8 *bits*	12 *bits*	16 *bits*	24 *bits*	8 *bits*	12 *bits*	16 *bits*	24 *bits*	8 *bits*	12 *bits*	16 *bits*	24 *bits*
C	ADSH	$C \to C$	0.9221	0.9555	0.9522	0.9521	0.9682	0.9894	0.9939	0.9865	0.8496	0.8843	0.9007	0.9184
		$C \to A$	0.3021	0.4613	0.5631	0.5503	0.6086	0.7704	0.8026	0.7193	0.3998	0.4205	0.3853	0.4151
		$A \to C$	0.4023	0.5350	0.6213	0.5443	0.6992	0.7805	0.7727	0.7639	0.4850	0.4715	0.4118	0.4318
		$A \to A$	0.1903	0.2829	0.3675	0.3255	0.4961	0.6410	0.6845	0.5632	0.3035	0.3300	0.3212	0.3468
	DSDH	$C \to C$	0.8172	0.8921	0.8947	0.8966	0.9565	0.9727	0.9821	0.9843	0.7007	0.6979	0.6978	0.7095
		$C \to A$	0.2879	0.3480	0.3423	0.3732	0.7712	0.8200	0.8493	0.8675	0.2258	0.3402	0.2803	0.2974
		$A \to C$	0.3723	0.4198	0.3973	0.4146	0.7873	0.8234	0.8148	0.8550	0.2578	0.3623	0.3032	0.3146
		$A \to A$	0.1939	0.2120	0.2065	0.2239	0.6425	0.6986	0.7001	0.7309	0.2265	0.3198	0.2858	0.3262
A	ADSH	$A \to A$	0.8124	0.8847	0.8791	0.8866	0.9550	0.9745	0.9718	0.9556	0.8531	0.8739	0.8852	0.8890
		$A \to C$	0.1243	0.1617	0.2339	0.3542	0.1006	0.1167	0.1563	0.1556	0.2527	0.3036	0.2631	0.2241
		$C \to A$	0.2322	0.2751	0.3100	0.3715	0.1920	0.1975	0.1802	0.2052	0.3438	0.4376	0.4116	0.3849
		$C \to C$	0.1181	0.1225	0.1395	0.1959	0.0956	0.0935	0.1123	0.1081	0.3173	0.3420	0.3134	0.3206
	DSDH	$A \to A$	0.6585	0.6940	0.7117	0.7081	0.9369	0.9787	0.9786	0.9779	0.6903	0.7061	0.6955	0.7037
		$A \to C$	0.1531	0.1373	0.1534	0.1502	0.1120	0.0912	0.1489	0.1202	0.2713	0.2389	0.2813	0.3549
		$C \to A$	0.2009	0.1764	0.1938	0.2038	0.1815	0.1731	0.1775	0.1798	0.3355	0.3034	0.3481	0.3884
		$C \to C$	0.1358	0.1186	0.1168	0.1194	0.1095	0.0935	0.1105	0.0989	0.3079	0.2967	0.3229	0.3565

Table 1: Test mAP results on three datasets.

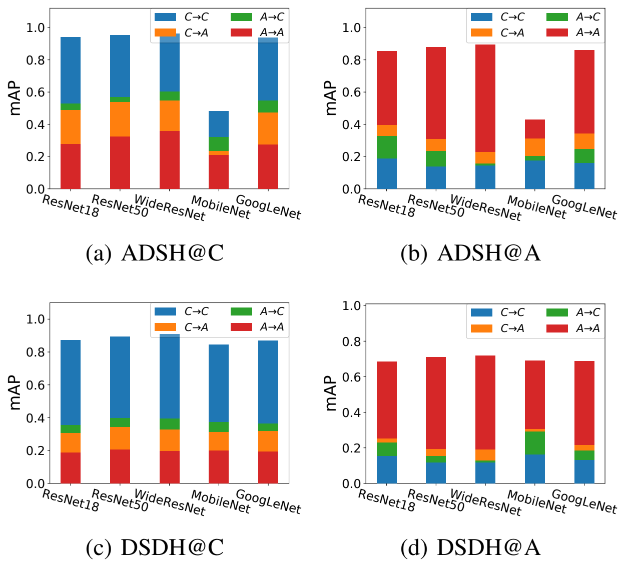


Figure 3: Transferability across different CNNs on CIFAR-10.

proposed PPL. For a quantitative analysis, on CIFAR-10, ADSH experiences average 50.8%, 44.0% and 70.7% performance degradations in adversarial data involved tasks, i.e., "$C{\to}A$", "$A{\to}C$" and "$A{\to}A$", respectively. While on DSDH, PPL achieves average fooling rates of 62.3%, 55.8% and 76.5%. Two methods undergo similar recession in performance when facing adversarial data on FashionMNIST dataset, with 54.0%, 47.3% and 62.9% average decreases on three tasks for ADSH, while 43.4%, 49.2% and 55.3% for DSDH. Although the protecting effects are not so significant on ImageNet, the proposed PPL still degrades the performance of two methods.

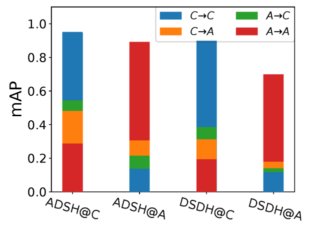
On the other hand, we utilize the adversarial data produced from the well-trained generator to train ADSH and DSDH, and then test their performance on various tasks. First, it can be observed that two models trained on adversarial data achieve excellent performance in the adversarial domain (i.e., "$A{\to}A$"), manifesting the adversarial perturbations do not adversely influence the normal uses of data. This verifies the practicability of the proposed PPL. Second, when facing clean data, two models lose their strong predictive power. More specifically, when using adversarial data to query clean data (i.e., "$A{\to}C$"), ADSH has only 29.2%, 13.7% and 42.4% normalized mean average precision compared to that in the adversarial domain (i.e., "$A{\to}A$") on CIFAR-10, ImageNet and FashionMNIST, respectively. The values for DSDH are 20.4%, 12.0% and 44.21%. In the "$C{\to}A$" task, our PPL reduces over 47% and 68% average performance of ADSH and DSDH on three datasets. Within clean domain (i.e., "$C{\to}C$"), ADSH merely achieves average 23.8% normalized mAP compared to that in the "$A{\to}A$" task. Similar results can also be seen on DSDH with the corresponding average value of 24.3%.

Transferability In realistic scenarios, retrieval models are usually built on different DNN architectures and data that we want to protect are from various sources, which implies the transferability is an important feature of the proposed method. To verify the generalization ability of the proposed PPL, we utilize the well-trained generator to test ADSH and DSDH built on different CNN backbones, i.e., ResNet18 (He et al. 2016), WideResNet (Zagoruyko and Komodakis 2016), Mobilenets (Howard et al. 2017) and GoogLeNet (Szegedy et al. 2015). The mAP results of 16 bits are reported in Figure 3, where "@C" and "@A" represents models are trained on clean data and adversarial data, respectively. At the same time, we use the generator trained on one dataset to fool models on another dataset. Considering space limitation and images on FashionMNIST are grey, only results between CIFAR-10 and ImageNet on two methods with code length as 16 are reported in Figure 4. From the figures, we can have the observation that the proposed PPL generalizes well across different networks and datasets, demonstrating its superiority and practicability.

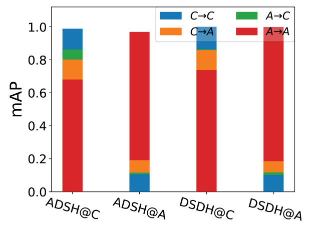
Ratio of Adversaries We consider the case when both clean data and adversarial data exist in the training set. The results of 16 bits with varying percentages of adversarial data are plotted in Figure 5, where we can see that with both data available for training, test models perform well on all tasks. This provides a possibility for authorized users to build models to perform search over adversarial data and clean data.

Effect of Distortion Here, we test the influence of the magnitude of the perturbations and plot the results with 16-bit code length in Figure 6. It can be observed that with ϵ increasing, the performance of two methods trained on one domain (e.g., clean domain) drops suddenly when facing data from the other domain (e.g., adversarial domain). It is quite normal as large distortions would inevitably enlarge the difference between adversarial data and its original counterparts. Besides, despite slight fluctuations, models trained on adversarial data still maintain performance when searching within adversaries ("$A \to A$") with increased distortions. This is in line with the (Ilyas et al. 2019) that adversarial perturbations arise as features instead of statistical anomalies.

Visualization Some generated adversarial examples are visualized in Figure 7, where the adversarial images are shown perceptually indistinguishable from the original clean images despite moderate distortions on FashionMNIST.

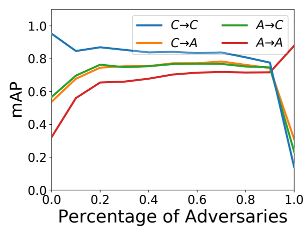


(a) ImageNet to CIFAR-10

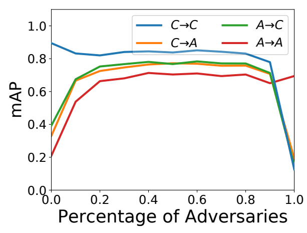


(b) CIFAR-10 to ImageNet

Figure 4: Transferability across different datasets.

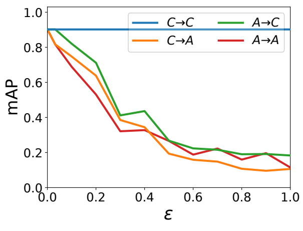


(a) ADSH

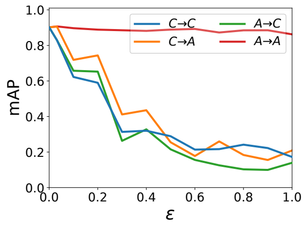


(b) DSDH

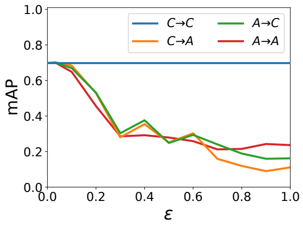
Figure 5: Ratio of adversaries on CIFAR-10.



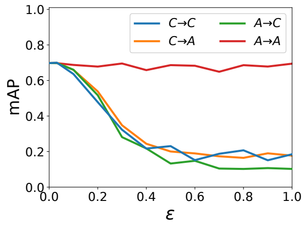
(a) ADSH@C



(b) ADSH@A

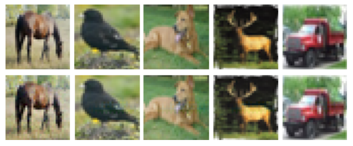


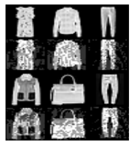
(c) DSDH@C



(d) DSDH@A

Figure 6: Effect of varying distoration on FashionMNIST.


(a) CIFAR - 10


(b) FashionMNIST

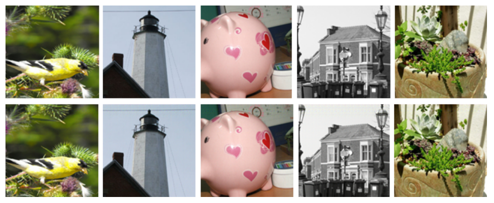

(c) ImageNet

Figure 7: Examples of clean data and its adversarial counterparts. First rows: original examples. Second rows: adversarial examples.

Conclusion

In this paper, we propose a data-centric Proactive Privacy-preserving Learning (PPL) algorithm to protect data privacy against unexpected or illegitimate uses of DNNs. The proposed method implements the protection mechanism before the data release, when the original data is transferred into the adversarial data via a generator without impairing the normal uses. The whole framework is trained by a two-player game between the generator and an imaginary victim retrieval model under various constraints, where both play against each other. A GRL module is inserted to enable a one step learning procedure. Extensive experiments demonstrate the superiority of the proposed method, including high protecting success rate and impressive transferability.

Acknowledgments

This work is partially supported by Australian Research Council Discovery Project (ARC DP190102353, CE200100025), and China Scholarship Council.

References

Babenko, A.; Slesarev, A.; Chigorin, A.; and Lempitsky, V. 2014. Neural codes for image retrieval. In *ECCV*, 584–599.

Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. Evasion attacks against machine learning at test time. In *ECML/PKDD*, 387–402.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: a large-scale hierarchical image database. In *CVPR*, 248–255.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *CVPR*, 9185–9193.

Ganin, Y.; and Lempitsky, V. 2015. Unsupervised domain adaptation by backpropagation. In *ICML*, 1180–1189.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* .

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *JMLR* 13(1): 723–773.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2017. Badnets: identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733* .

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* .

Ilyas, A.; Santurkar, S.; Tsipras, D.; Engstrom, L.; Tran, B.; and Madry, A. 2019. Adversarial examples are not bugs, they are features. In *NeurIPS*, 125–136.

Jiang, Q.-Y.; and Li, W.-J. 2018. Asymmetric deep supervised hashing. In *AAAI*, 3342–3349.

Kang, G.; Jiang, L.; Yang, Y.; and Hauptmann, A. G. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*, 4893–4902.

Krizhevsky, A.; et al. 2009. Learning multiple layers of features from tiny images. *TR* .

Li, Q.; Sun, Z.; He, R.; and Tan, T. 2017. Deep supervised discrete hashing. In *NeurIPS*, 2482–2491.

Li, W.-J.; Wang, S.; and Kang, W.-C. 2016. Feature learning based deep supervised hashing with pairwise labels. In *IJCAI*, 1711–1717.

Liu, H.; Wang, R.; Shan, S.; and Chen, X. 2016. Deep supervised hashing for fast image retrieval. In *CVPR*, 2064–2072.

Liu, X.; Yu, G.; Domeniconi, C.; Wang, J.; Ren, Y.; and Guo, M. 2019. Ranking-based deep cross-modal hashing. In *AAAI*, 4400–4407.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2574–2582.

Mopuri, K. R.; Garg, U.; and Babu, R. V. 2017. Fast feature fool: a data independent approach to universal adversarial perturbations. *arXiv preprint arXiv:1707.05572* .

Papernot, N.; McDaniel, P.; and Goodfellow, I. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* .

Shafahi, A.; Huang, W. R.; Najibi, M.; Suciu, O.; Studer, C.; Dumitras, T.; and Goldstein, T. 2018. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NeurIPS*, 6103–6113.

Shen, F.; Xu, Y.; Liu, L.; Yang, Y.; Huang, Z.; and Shen, H. T. 2018. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *TPAMI* 40(12): 3034–3044.

Shen, H. T.; Jiang, S.; Tan, K.-L.; Huang, Z.; and Zhou, X. 2009. Speed up interactive image retrieval. *VLDB* 18(1): 329–343.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*, 1–9.

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

Tramèr, F.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453* .

Wang, Z.; Zhang, Z.; Luo, Y.; Huang, Z.; and Shen, H. T. 2020. Deep collaborative discrete hashing with semantic-invariant structure construction. *TMM* .

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

Yang, Y.; Luo, Y.; Chen, W.; Shen, F.; Shao, J.; and Shen, H. T. 2016. Zero-shot hashing via transferring supervised knowledge. In *ACM MM*, 1286–1295.

Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146* .

Zhang, Z.; Xie, G.-s.; Li, Y.; Li, S.; and Huang, Z. 2019. SADIH: semantic-aware discrete hashing. In *AAAI*, 5853–5860.

Zhou, X.; Shen, F.; Liu, L.; Liu, W.; Nie, L.; Yang, Y.; and Shen, H. T. 2020. Graph convolutional network hashing. *TCYB* 50(4): 1460–1472.

Zhu, X.; Huang, Z.; Cheng, H.; Cui, J.; and Shen, H. T. 2013. Sparse hashing for fast multimedia search. *TOIS* 31(2): 1–24.