Private Model Compression via Knowledge Distillation

Ji Wang,[1] **Weidong Bao,**[1] **Lichao Sun,**[2] **Xiaomin Zhu,**[1,3] **Bokai Cao,**[4] **Philip S. Yu**[2,5]
[1]College of Systems Engineering, National University of Defense Technology, Changsha, China
[2]Department of Computer Science, University of Illinois at Chicago, Chicago, USA
[3]State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha, China
[4]Facebook Inc., Menlo Park, USA
[5]Institute for Data Science, Tsinghua University, Beijing, China
{wangji,wdbao,xmzhu}@nudt.edu.cn, lsun29@uic.edu, caobokai@fb.com, psyu@uic.edu

Abstract

The soaring demand for intelligent mobile applications calls for deploying powerful deep neural networks (DNNs) on mobile devices. However, the outstanding performance of DNNs notoriously relies on increasingly complex models, which in turn is associated with an increase in computational expense far surpassing mobile devices' capacity. What is worse, app service providers need to collect and utilize a large volume of users' data, which contain sensitive information, to build the sophisticated DNN models. Directly deploying these models on public mobile devices presents prohibitive privacy risk. To benefit from the on-device deep learning without the capacity and privacy concerns, we design a private model compression framework RONA. Following the knowledge distillation paradigm, we jointly use hint learning, distillation learning, and self learning to train a compact and fast neural network. The knowledge distilled from the cumbersome model is adaptively bounded and carefully perturbed to enforce differential privacy. We further propose an elegant query sample selection method to reduce the number of queries and control the privacy loss. A series of empirical evaluations as well as the implementation on an Android mobile device show that RONA can not only compress cumbersome models efficiently but also provide a strong privacy guarantee. For example, on SVHN, when a meaningful $(9.83, 10^{-6})$-differential privacy is guaranteed, the compact model trained by RONA can obtain $20\times$ compression ratio and $19\times$ speed-up with merely 0.97% accuracy loss.

Introduction

Recent years have witnessed impressive breakthroughs of deep learning in various areas. Thanks to the large volume of data and the easy availability of computational resources, customized deep learning models attain unprecedented performance that beats many records of traditional machine learning algorithms.

The significant progress in deep learning, on the other hand, is notorious for its dependence on cumbersome models which require massive data to learn their millions of parameters. This major drawback restricts the large-scale deployment of deep learning applications, especially the deployability of deep neural network on mobile devices such as smartphones, wearable devices, and medical monitors (Cao et al. 2017; Sun et al. 2017). Mobile application service providers are facing a series of challenges to widely adopt DNNs in their mobile apps.

Capacity bottleneck. In spite of the great advances of mobile chips and mobile batteries, the limited-capacity nature of mobile devices still imposes the intrinsic bottleneck making resource-demanding applications remain off bounds (Wang et al. 2018). The modern DNNs with millions of parameters often require prohibitive runtime to execute on computationally limited devices. What is worse, the massive floating point operations during the execution aggravate the burden of processing chips and easily dominate the whole system energy consumption, which widens the chasm between DNNs and their deployability on mobile devices. It is difficult to directly adopt powerful DNNs on mobile systems so far (Lee 2017).

Data privacy and intellectual piracy concerns. Developing a strong predictive DNN needs abundant data. The more data accessible, the more effective and powerful a DNN will be. In practice, app service providers usually collect and utilize a large volume of data from users, which often contains sensitive information, to build their sophisticated DNN models. Directly releasing these models trained by users' data presents potential privacy issues because the adversary can recover sensitive information encoded in the public models (Abadi et al. 2016). It is illegal to share individuals' data or models directly in many domains like medicine (Claerhout and DeMoor 2005). Apart from data privacy, releasing DNN models may invade app service providers' right due to intellectual piracy. It is sometimes not appealing for app service providers to share their valuable and highly tuned models (Osia et al. 2017).

In order to deploy efficient DNNs on mobile devices, academia and industry put forward a number of model compression methods among which knowledge distillation plays a key role (Bucilua, Caruana, and Niculescu-Mizil 2006). In knowledge distillation, the knowledge embedded in the cumbersome model, known as the teacher model, is distilled to guide the training of a smaller model called the student model. The student model has a different architecture and fewer parameters, but can achieve comparable performance by mimicking the behavior of the cumbersome model. Other compression methods like quantization and low-rank factorization (Han, Mao, and Dally 2016;

Howard et al. 2017) are complementary to knowledge distillation and can also be used to further reduce the size of student models, which is beyond the scope of this paper. Despite the encouraging compression rate, the privacy concern has not been fully resolved by the current knowledge distillation methods yet.

In this paper, we introduce the rigorous standard of differential privacy (Dwork 2011) and design a p**R**ivate m**O**del compressio**N** fr**A**mework, named RONA, to overcome the aforementioned challenges and promote the adoption of deep learning in mobile applications.

Both model compression and data privacy are considered in the proposed framework. We assume that the app service provider has trained a powerful cumbersome model based on the sensitive data and the public data. Following the knowledge distillation paradigm, RONA uses only the public data to train the small student model whose feature representations are encouraged to be similar to those of the cumbersome model. Then the small student model is released to mobile application users while the cumbersome model as well as the sensitive data are retained by the app service provider. Intuitively, the privacy is preserved because the training of the student model does not depend on the sensitive data; neither the cumbersome model nor the sensitive data is exposed to the public or accessible to the adversary. Mere intuition, however, is not sufficient. To provide provable privacy guarantee, we carefully perturb the knowledge distilled from the cumbersome model to satisfy the standard of differential privacy.

It is a nontrivial problem to jointly compress large DNNs, preserve privacy, and control model performance loss. A set of novel methods are presented to solve this problem. Our main contributions are four-fold:

- **A framework promoting deep learning in mobile applications**. We take the key constraints of adopting DNNs in mobile applications, *i.e.*, privacy, performance, and overhead, into consideration, and design a privacy-preserving framework for training compact neural networks via knowledge distillation[1]. To the best of our knowledge, the proposed RONA is the first framework that applies the knowledge distillation to model compression with meaningful privacy guarantee.
- **A differentially private knowledge distillation**. To theoretically guarantee privacy, we propose a new mechanism to perturb the knowledge distillation from the sense of differential privacy. Different from the existing sample-by-sample query mode, our proposed mechanism makes queries in batch mode to reduce the number of queries. The batch loss responded by the teacher model is clipped by the adaptive norm bound and then carefully perturbed to preserve privacy.
- **A query sample selection method for knowledge distillation**. The privacy loss depends on the number of queries. To control the student model's access to the teacher model, the number of samples used during knowledge distillation should be reduced. Hence, we present a query sample selection method to select a subset of samples such that the knowledge distilled over the subset is competitive over the whole samples.
- **Thorough empirical evaluation**. We evaluate the proposed RONA by using three standard benchmarks that are widely used in knowledge distillation works. The results demonstrate the effectiveness of the above novel methods, bringing significant improvement in training small models with rigorous privacy guarantee. Our code is open-sourced at https://github.com/jwanglearn/Private_Compress.

[1]This paper uses the multi-class image recognition as the application scenario due to its wide usage in mobile apps.

Preliminary and Related Work

Deep neural network compression. To squeeze DNNs into mobile devices, DNN compression attracts intense attention. The compression methods can be broadly classified into four categories: parameter sharing, network pruning, low-rank factorization, and knowledge distillation (Han, Mao, and Dally 2016; Howard et al. 2017; Chen et al. 2017). The former three methods mainly attempt to reduce the size of a given model, without significantly changing the architecture of the model. The last one, knowledge distillation, uses the knowledge captured by the cumbersome model to guide the training of a smaller model in a teacher-student paradigm (Bucilua, Caruana, and Niculescu-Mizil 2006). Hinton et al. (Hinton, Vinyals, and Dean 2014) used the class probabilities generated by the teacher as a soft target to train the student model. Romero et al. (Romero et al. 2015) extended this work by training the student to mimic the teacher's intermediate representation. Based on these two works, a framework for compressing object detection models was designed by Chen et al. (Chen et al. 2017), trying to solve the class imbalance problem. Few existing model compression methods took the privacy issue into consideration. The teacher model can be queried as many times as necessary during training, which is infeasible if we want to preserve privacy.

Differentially private deep learning. Due to the critical need of respecting privacy, privacy-preserving data analysis has become an emerging topic of interests. One state-of-the-art privacy standard is differential privacy (Dwork 2011) which provides provable privacy guarantee.

Definition 1 *(Dwork 2011) A randomized mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private, iff for any adjacent input d and d', and any output S of $\mathcal{M}$,*

$$\Pr[\mathcal{M}(d) = S] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(d') = S] + \delta. \quad (1)$$

Typically, d and d' are adjacent inputs when they are identical except for only one data item. The parameter ϵ denotes the privacy budget (Dwork 2011), controlling the privacy loss of $\mathcal{M}$. A smaller value of ϵ enforces a stronger privacy guarantee.

For a deterministic function f, the (ϵ, δ)-differential privacy is generally enforced by injecting random noise calibrated to the f's sensitivity Δf, $\Delta f = \max_{d,d'} \|f(d) - f(d')\|$. For example, the Gaussian mechanism is given by,

Theorem 1 *(Dwork and Roth 2014) Suppose function f with L2 norm sensitivity $\Delta_2 f$, a randomized mechanism*

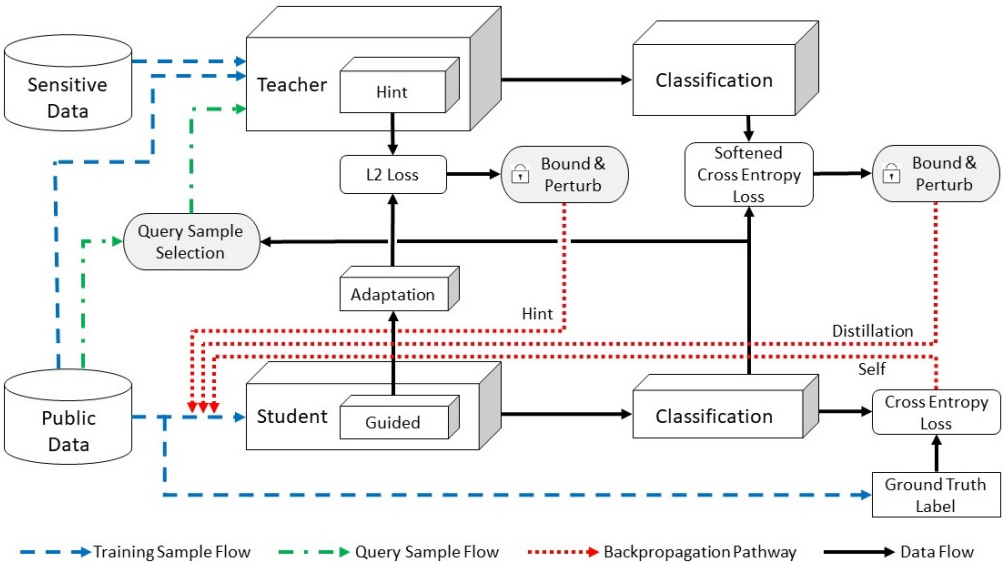


Figure 1: The overview of RONA. RONA is implemented by the mobile app service providers. Only the student model is released to the public while the sensitive data and the teacher model are retained by the mobile app service provider.

$\mathcal{M}_f(d)$:

$$\mathcal{M}_f(d) = f(d) + \mathcal{N}(0, \Delta_2 f^2 \sigma^2), \tag{2}$$

where $\mathcal{N}(0, \Delta_2 f^2 \sigma^2)$ *is a random variable sampled from the Gaussian distribution with mean 0 and standard deviation* $\Delta_2 f \sigma$. $\mathcal{M}_f(d)$ *is* (ϵ, δ)*-differentially private if* $\sigma \geq \sqrt{2\ln(1.25/\delta)}/\epsilon$ *and* $\epsilon < 1$.

Differential privacy provides guaranteed privacy which cannot be compromised by any algorithm (Dwork 2011). It is increasingly adopted as the standard notion of privacy (Beimel et al. 2014). Abadi et al. (Abadi et al. 2016) presented a differentially private SGD and designed a new technique to track the privacy loss. Papernot et al. (Papernot et al. 2017; Papernot et al. 2018) proposed a general framework for private knowledge transfer in the sample-by-sample mode where the student was trained to predict an output generated by noisy voting among the teachers. This framework only used the output label to train the student and cannot be used to compress DNNs efficiently. Apart from private training, Triastcyn et al. (Triastcyn and Faltings 2018) added a Gaussian noise layer in the discriminator of GAN to generate differentially private data. Recently, Wang et al. (Wang et al. 2018) designed a private inference framework across mobile devices and cloud servers to free mobile devices from complex inference tasks. However, this framework heavily depended on the network accessibility. Few current works applied differential privacy and model compression to enabling on-device deep learning as well as preserving privacy.

The Proposed Framework

The framework RONA is presented in this section. We first give the overview of RONA. Then we detail three key modules: (1) the model compression based on knowledge distillation, (2) the differentially private knowledge perturbation, and (3) the query sample selection.

Overall Structure

The overview of RONA is given in Fig. 1. To better capture the knowledge embedded in the cumbersome teacher model, we jointly use the hint learning (Romero et al. 2015), the distillation learning (Hinton, Vinyals, and Dean 2014), and the self learning to train a small student model. Meanwhile, both the hint loss and the distillation loss are carefully bounded and perturbed by random noises that are consistent with differential privacy. Since more queries to the teacher model incur higher privacy loss, an elegant query sample selection method is designed in RONA to select a subset of samples from the entire public samples.

The sensitive data is only used to train the complex teacher model which is not released to the public. It is obvious that the sensitive data are insulated from the explicit invasion of privacy as the student model has no access to it. Further, the information generated by the teacher model, *i.e.*, the hint loss and the distillation loss, are perturbed by additional noises. All the information relating to the sensitive data and the teacher model are isolated or well protected before releasing.

Model Compression

In order to learn a compact and fast DNN model that can be adopted in mobile applications, we propose a stage-wise knowledge distillation method. This method enables the small student model to capture not only the information in ground truth labels but also the information distilled from the cumbersome teacher model.

Hint learning stage: We start by teaching the student model how to extract features from the input data. The intermediate representation of the teacher model is used as a

Algorithm 1: Compact Student Model Training

Parameter : Hint learning epochs T_h; Distillation learning epochs T_d; Self learning epochs T_s; Iterations R; Hint and distillation learning batch size S; Self learning batch size S'.

for $t \leftarrow 0$ **to** $T_h - 1$ **do**
  $\mathbf{x}_q \leftarrow$ `random_select`$(\mathbf{x}_p)$;
  for $i \leftarrow 0$ **to** $|\boldsymbol{x}_q|/S$ **do**
    Sample a batch $\mathbf{x}_q^{(i)}$ of size S from $\mathbf{x}_q$;
    Compute hint loss $\mathcal{L}_{hint}^{(i)}$ over $\mathbf{x}_q^{(i)}$ by Eq. (3);
    $\widetilde{\mathcal{L}}_{hint}^{(i)} \leftarrow$ `privacy_sanitize`$(\mathcal{L}_{hint}^{(i)})$;
    Backpropagate by $\widetilde{\mathcal{L}}_{hint}^{(i)}$ to update Θ_g and Θ_a;
for $r \leftarrow 0$ **to** $R - 1$ **do**
  for $t \leftarrow 0$ **to** $T_s - 1$ **do**
    for $i \leftarrow 0$ **to** $|\boldsymbol{x}_p|/S'$ **do**
      Sample a batch $\mathbf{x}_p^{(i)}$ of size S' from $\mathbf{x}_p$;
      Compute self loss $\mathcal{L}_{self}^{(i)}$ over $\mathbf{x}_p^{(i)}$ by Eq. (5);
      Backpropagate by $\mathcal{L}_{self}^{(i)}$ to update Θ_s;
  for $t \leftarrow 0$ **to** $T_d - 1$ **do**
    $\mathbf{x}_q \leftarrow$ `query_select`$(\mathbf{x}_p, \Theta_s)$;
    for $i \leftarrow 0$ **to** $|\boldsymbol{x}_q|/S$ **do**
      Sample a batch $\mathbf{x}_q^{(i)}$ of size S from $\mathbf{x}_q$;
      Compute distill loss $\mathcal{L}_{distill}^{(i)}$ over $\mathbf{x}_q^{(i)}$ by Eq. (4);
      $\widetilde{\mathcal{L}}_{distill}^{(i)} \leftarrow$ `privacy_sanitize`$(\mathcal{L}_{distill}^{(i)})$;
      Backpropagate by $\widetilde{\mathcal{L}}_{distill}^{(i)}$ to update Θ_s;

hint to guide the training of the student model. Analogously, a hidden layer of the student model is chosen as the guided layer that learns from the teacher's guidance. We train the student model from the first hidden layer up to the guided layer by minimizing the L2 loss function:

$$\mathcal{L}_{hint}(\mathbf{x}_q, \mathbf{z}_h; \Theta_g, \Theta_a) = \frac{1}{2}\|r(g(\mathbf{x}_q; \Theta_g); \Theta_a) - \mathbf{z}_h\|^2, \quad (3)$$

where $g(\cdot; \Theta_g)$ represents the student model up to the guided layer with parameter Θ_g, $\mathbf{x}_q$ denotes the query samples, and $\mathbf{z}_h$ is the output of the teacher's hint layer over the query samples. As the scale of the guided layer is usually different from that of the hint layer, we introduce an adaptation layer on the top of the guided layer to match the scale of the guided and hint layers. The adaptation layer is represented by $r(\cdot; \Theta_a)$ with parameter Θ_a learned during the hint learning. If both guided and hint layers are fully connected layers, we add a fully connected layer as the adaptation layer. If both guided and hint layers are convolutional, we add 1×1 convolutions instead to reduce the number of parameters.

Hint learning is introduced to teach the student how to extract general features. It makes the student model lack flexibility to choose a higher hidden layer as the guided layer. So, we select the student's middle layer as the guided layer in our case.

Distillation and self learning stage: We train the whole student model by using the distillation and self learning in this stage. Let $\mathbf{z}_t$ be the output of the teacher's final hidden layer, also called logits, over the query samples, we use the soften probability (Hinton, Vinyals, and Dean 2014) $\mathbf{P}_t^\tau$ as the knowledge: $\mathbf{P}_t^\tau = softmax(\mathbf{z}_t/\tau)$, where τ is the temperature parameter that is normally set as 1. A higher τ can make the teacher model generate soften probabilities such that the classes whose normal probabilities are near zero will not be ignored. The soften probability contains the knowledge about the latent relationship between different classes. The student model is trained to learn this knowledge by minimizing the distillation loss over the query samples:

$$\mathcal{L}_{distill}(\mathbf{x}_q, \mathbf{P}_t^\tau; \Theta_s) = \mathcal{H}(\mathbf{P}_s^\tau, \mathbf{P}_t^\tau; \Theta_s), \quad (4)$$

where $\mathcal{H}$ is the cross-entropy, and Θ_s is the parameters of the student model. Here, $\mathbf{P}_s^\tau$ is the student's soften probability over the query samples $\mathbf{x}_q$, $\mathbf{P}_s^\tau = softmax(\mathbf{z}_s/\tau)$, where $\mathbf{z}_s$ is the student's logits.

Different from the classical use of knowledge distillation, we introduce the self learning process where the ground truth labels of all public samples are used to train the student model by minimizing the self loss:

$$\mathcal{L}_{self}(\mathbf{x}_p, \mathbf{y}_p; \Theta_s) = \mathcal{H}(\mathbf{P}_s, \mathbf{y}_p; \Theta_s), \quad (5)$$

where $\mathbf{x}_p$ and $\mathbf{y}_p$ are the public samples and the ground truth labels, respectively, and $\mathbf{P}_s$ is the normal probability generated by the student model with $\tau = 1$.

For the privacy-preserving reason, the number of distillation-learning epochs should be controlled (detailed in the next section). Nonetheless, the self learning does not use any information relating to the sensitive data, and hence it does not aggregate the privacy loss. The number of self-learning epochs can be arbitrarily large. It is inappropriate to combine the distillation loss and the self loss together as a general loss for training. To jointly apply the two learning methods, we mimic the way how a real student learns from her teacher. The student model first learns by itself to minimize the self loss. Then, it selects some samples to query the teacher model, and learns from the teacher by minimizing the distillation loss. This procedure repeats until the convergence or exceeding the privacy budget. Our experimental results show that the self learning can not only accelerate the training of the student model but also allow the distillation learning to avoid local minima.

Algorithm 1 presents the two-stage knowledge distillation. The student model queries the teacher model in a batch-by-batch mode. During the hint learning, the student model learns to extract general features. Hence, we select the query samples from the public samples randomly rather than using the proposed sample selection method.

Privacy Protection

To enforce theoretical privacy guarantee, we inject random perturbation into the information that is related with the sensitive data and is used by the student model, *i.e.*, the hint loss and the distillation loss. Algorithm 2 outlines the privacy-preserving function `privacy_sanitize`. For each batch loss $\mathcal{L}^{(i)}$, we first bound the batch loss by a threshold B, ensuring that the sensitivity is not larger than B; then we inject Gaussian noise into the bounded batch loss.

Algorithm 2: Function Privacy Sanitize

Input: Batch loss $\mathcal{L}^{(i)}$.
Parameter : Noise scale σ; Bound threshold B.
1 $\bar{\mathcal{L}}^{(i)} \leftarrow \mathcal{L}^{(i)} / \max\left(1, \frac{\left\|\mathcal{L}^{(i)}\right\|_2}{B}\right)$;
2 $\widetilde{\mathcal{L}}^{(i)} \leftarrow \bar{\mathcal{L}}^{(i)} + \mathcal{N}(0, \sigma^2 B^2 \mathbf{I})$;
Output: Sanitized batch loss $\widetilde{\mathcal{L}}^{(i)}$.

It is hard to estimate the sensitivity of the batch loss over the sensitive data. Therefore, we clip the max value of $\left\|\mathcal{L}^{(i)}\right\|_2$ within a given bound as shown in Line 1 of Algorithm 2. The value of $\left\|\mathcal{L}^{(i)}\right\|_2$ is preserved if $\left\|\mathcal{L}^{(i)}\right\|_2 \leq B$, whereas it is scaled down to B if $\left\|\mathcal{L}^{(i)}\right\|_2 > B$. After clipping, the sensitivity of $\bar{\mathcal{L}}^{(i)}$ is B.

An overly large B will incur excessive noise, while too small B will lead to over truncation of the batch loss, both causing low utility of the sanitized batch loss. To solve this problem, we propose to use an adaptive norm bound. Specially, we train an auxiliary teacher model based on the public data. We constantly monitor the auxiliary batch loss between the auxiliary teacher model and the student model during training and set the average value of the auxiliary batch loss as the norm bound. In this manner, the norm bound B changes adaptively during training. The empirical study shows considerable performance improvement brought by such an adaptive norm bound. As the norm bound is independent with the sensitive data, no privacy budget would be consumed by clipping the batch loss.

Gaussian noise is then added into the bounded batch loss to preserve privacy. According to Theorem 1, this randomized mechanism enforces (ϵ, δ)-differential privacy per query if we set σ as $\sqrt{2\ln(1.25/\delta)}/\epsilon$. During the training of the student model, the teacher model is queried $T = (T_h + RT_d)|\mathbf{x}_q|/S$ times. Using moments accountant (Abadi et al. 2016), we have the following theorem:

Theorem 2 *Given $\epsilon < c_1 T$, $\delta > 0$, where c_1 is a constant. Algorithm 1 can achieve (ϵ, δ)-differential privacy by setting σ as:*

$$\sigma \geq c_2 \frac{\sqrt{T \ln(1/\delta)}}{\epsilon}, \tag{6}$$

where c_2 is a constant.

Due to the page limitation, we omit the proof here. Theorem 2 indicates that a larger value of T incurs a larger privacy budget ϵ, namely more privacy loss, when σ is fixed. To provide stronger protection, the value of T should be controlled. Therefore, instead of querying the teacher over all public samples, we select the subset $\mathbf{x}_q$ as the query samples. Nonetheless, it is obvious that the downsampling of public samples has an adverse impact on knowledge distillation. To alleviate this impact, we design a novel query sample selection method to select the critical samples in the next section.

Query Sample Selection

The sample selection problem can be categorized as the active learning problem. Different from the traditional setting where the samples are chosen one-by-one to query, our proposed student-teacher query works in a batch mode. We attempt to select a set of query samples such that the distilled knowledge over the query samples and that over the whole public samples are as close as possible. Formally, we try to minimize the difference between the distillation loss over the query samples and that over the whole public samples:

$$\min_{\mathbf{x}_q : \mathbf{x}_q \subset \mathbf{x}_p} \left| \mathcal{L}_{distill}(\mathbf{x}_p, \mathbf{P}_t^\tau) - \mathcal{L}_{distill}(\mathbf{x}_q, \mathbf{P}_t^\tau) \right|. \tag{7}$$

Because we have no prior knowledge of $\mathbf{P}_t^\tau$, the above optimization objective is not tractable. Instead, we try to optimize the upper bound of this objective as given below.

Theorem 3 *Given the public samples $\boldsymbol{x}_p$ and the query samples $\boldsymbol{x}_q$. If $\boldsymbol{x}_q$ is λ cover of $\boldsymbol{x}_p$, $L_{distill}(\boldsymbol{x}_p, \boldsymbol{P}_t^\tau) = A$, we have,*

$$\begin{aligned} &|\mathcal{L}_{distill}(\boldsymbol{x}_p, \boldsymbol{P}_t^\tau) - \mathcal{L}_{distill}(\boldsymbol{x}_q, \boldsymbol{P}_t^\tau)| \\ &\leq \mathcal{O}(\lambda) + \mathcal{O}(\sqrt{\frac{1}{|\boldsymbol{x}_p|}}) + \mathcal{O}(A). \end{aligned} \tag{8}$$

The proof is omitted here due to the page limitation. In Theorem 3, "$\mathbf{x}_q$ is λ cover of $\mathbf{x}_p$" indicates that the whole $\mathbf{x}_p$ can be covered by a group of spheres[2] centered at each sample in $\mathbf{x}_q$ with radius λ. In the right-hand side (RHS) of Eq. (8), $\mathcal{O}(\sqrt{1/|\mathbf{x}_p|}) + \mathcal{O}(A)$ is independent with $\mathbf{x}_q$. Hence, we can minimize λ to control the RHS of Eq. (8). Now the optimization of (7) is converted into: $\min_{\mathbf{x}_q : \mathbf{x}_q \subset \mathbf{x}_p} \lambda$. This problem is equivalent to the minimax location problem (Korupolu, Plaxton, and Rajaraman 1998),

$$\min_{\mathbf{x}_q : \mathbf{x}_q \subset \mathbf{x}_p} \max_{x_i \in \mathbf{x}_q} \min_{x_j \in \mathbf{x}_q, x_j \neq x_i} \mathcal{D}(x_i, x_j | \Theta_s), \tag{9}$$

where $\mathcal{D}(x_i, x_j|\Theta_s)$ denotes the distance between two samples. The KL-divergence between the output probabilities is used as the distance. We use a 2-OPT greedy algorithm to solve this problem (Korupolu, Plaxton, and Rajaraman 1998). Algorithm 3 outlines the function for selecting query samples.

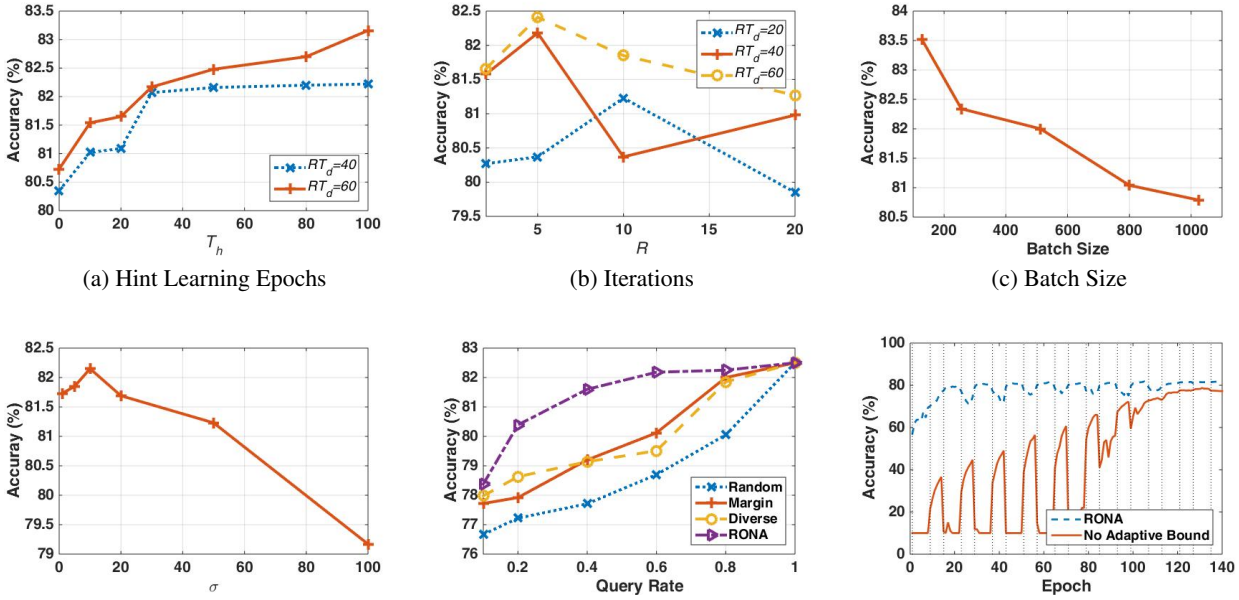
Algorithm 3: Function Query Select

Input: Public samples $\mathbf{x}_p$; Student parameters Θ_s.
Parameter : Number of query samples N_q.
1 Initialize $\mathbf{x}_q$ with a sample randomly chosen from $\mathbf{x}_p$;
2 **for** $n \leftarrow 0$ **to** $N_q - 2$ **do**
3 $\quad x_{sel} \leftarrow \arg\max_{x_i \in \mathbf{x}_p - \mathbf{x}_q} \min_{x_j \in \mathbf{x}_q} \Lambda(x_i, x_j | \Theta_s)$;
4 $\quad \mathbf{x}_q \leftarrow \mathbf{x}_q \cup \{x_{sel}\}$;
Output: Query samples $\mathbf{x}_q$.

Experimental Evaluation

The framework RONA is evaluated based on three popular image datasets: MNIST (LeCun et al. 1998), SVHN (Netzer et al. 2011), and CIFAR-10 (Krizhevsky and Hinton 2009). We first use CIFAR-10 to examine the performance impact of different parameters and the effectiveness

[2]Here we use the concepts in 3D space to make it easy-to-understand. The distance is defined in the feature space of data.



(a) Hint Learning Epochs (b) Iterations (c) Batch Size

(d) Noise Scale (e) # of Query Samples (f) Testing Accuracy

Figure 2: Performance impact of parameters.

of proposed techniques in RONA. Then we verify privacy protection and compression performance based on MNIST, SVHN, and CIFAR-10. The experiment details are reported in the GitHub repository together with the codes.

Effect of Parameters

We use CIFAR-10 in this group of experiments. CIFAR-10 contains 50K training samples belonging to 10 classes. We randomly choose 80% training samples as the public data while the reset 20% as the sensitive data. We preprocess the dataset by normalizing each sample. The widely used convolutional deep neural network, Conv-Large (Laine 2017; Park et al. 2017), is pretrained as the cumbersome teacher model on both public data and sensitive data. A modified Conv-Small network is used as the compact student model that will be trained based on RONA. The performance of the compact student model is affected by multiple parameters. We examine them individually, keeping the others constant, to show their effects.

Hint learning epochs. It can be observed from Fig. 2(a) that the accuracy of the student model increases when the hint learning epoch ascends. But the increase diminishes as the hint learning epoch becomes large, especially when RT_d, *i.e.*, the total epochs of distillation learning, is small. As the hint learning consumes the privacy budget, it is not appropriate to set an overly large value of hint learning epoch.

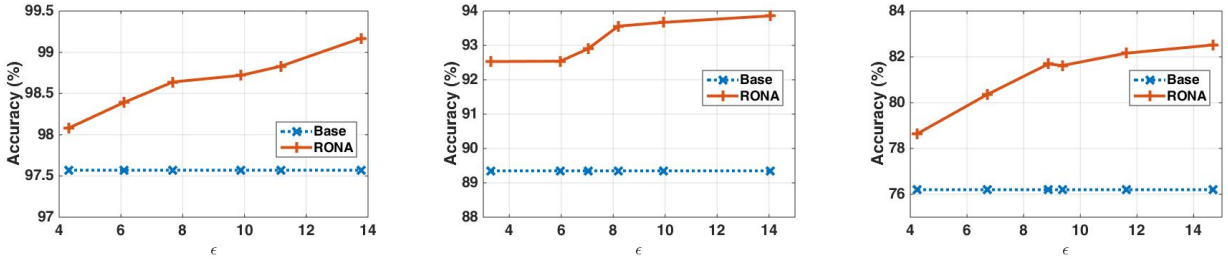
Iterations for distillation learning. The total epochs of distillation learning are determined by the rounds of iterations R and the epochs per iteration T_d. Generally, as shown in Fig. 2(b), a larger value of RT_d brings a more effective student model because the student model can learn more knowledge from the teacher. When RT_d is fixed, R should be set as a moderate value.

Batch size. Fig. 2(c) shows that the student's performance descends with the increase of batch size. In RONA, the student queries the teacher in a batch-by-batch mode. A small value of batch size indicates that the teacher would be queried more times, and thus the privacy loss would be high. To achieve a balance between performance and privacy, we set the batch size as 512 in our experiments.

Noise scale. The student model benefits from the additional noise when the noise scale is moderate as shown in Fig. 2(d). As a DNN usually suffers from the overfitting problem, the norm bound and additional noise act as regularization roles during training. Even when the noise scale is relatively large ($\sigma = 20$), the accuracy degradation is less than 1%. This property is encouraging as more noise can be injected to provide a stronger privacy guarantee per query.

Number of query samples. The experimental results in Fig. 2(e) show that the student's performance rises when more query samples are available. Nonetheless, more query samples mean a higher privacy loss. The query sample selection method proposed in our work can achieve decent performance by only using 20% public samples as the query samples. We compare it with three other selection methods: random, margin (Wang et al. 2017), and diverse (Wang and Ye 2013). Fig. 2(e) demonstrates the superiority of our proposed query sample selection method.

Adaptive norm bound. We plot the testing accuracy during the distillation and self learning stage in Fig. 2(f). Compared with training with the pre-set norm bound, the adaptive norm bound method brings significant performance improvement. It greatly accelerates the training process, obtaining a higher accuracy with much fewer query epochs



(a) MNIST (b) SVHN (c) CIFAR

Figure 3: Accuracy vs. privacy budget ϵ. Base denotes the student model trained without querying the teacher model.

Table 1: Performance of Masking Specific Samples

	Base	RONA		
		$\epsilon = 5.2$	$\epsilon = 8.7$	$\epsilon = 29.8$
Overall	79.87	80.93	88.48	90.35
Unseen Classes	0.00	9.59	46.75	53.43

(*i.e.*, less privacy loss). Besides, we can find that the self learning contributes to the training acceleration as well. At the beginning of this stage, the accuracy is quickly improved by the self learning without consuming privacy budget.

Privacy Protection

We verify the privacy protection on MNIST, SVHN, and CIFAR-10. MNIST and SVHN are digit image datasets consisting of 60K and 73K training samples, respectively. We randomly choose 40% SVHN training samples and MNIST training samples as the public data. For MNIST, we use a modified Conv-Small network as the teacher. For SVHN, we use the Conv-Middle network as the teacher.

RONA achieves accuracies of 98.64% and 92.90% on MNIST and SVHN with $(7.68, 10^{-5})$ and $(7.03, 10^{-6})$ differential privacy. It outperforms the results in (Papernot et al. 2017) which achieved 98.10% and 90.66% accuracy on $(8.03, 10^{-5})$ guaranteed MNIST and $(8.19, 10^{-6})$ guaranteed SVHN. It is comparable with the latest results in (Papernot et al. 2018), achieving 98.5% and 91.6% accuracy on $(1.97, 10^{-5})$ guaranteed MNIST and $(4.96, 10^{-6})$ guaranteed SVHN, which however used better and larger baseline networks. On CIFAR-10, we obtain 81.69% accuracy with $(8.87, 10^{-5})$ privacy, which outperforms 73% accuracy with $(8.00, 10^{-5})$ privacy in (Abadi et al. 2016). RONA's performance is even better than the latest cloud-based solution that achieved 79.52% accuracy on CIFAR-10 (Wang et al. 2018).

We verify the performance with different privacy budgets on three datasets. Fig. 3 demonstrates that the accuracies on three datasets generally rise with the increase of privacy budget. Thanks to the techniques proposed in this work, the student can still benefit from the knowledge distillation even when the knowledge is protected by a strong privacy.

In the above experiments, we select the public data randomly from the original training samples. We further test RONA in a much tougher condition on MNIST where we regard all training samples of the digits 6 and 9 as sensitive data. This data masking mimics a possible contingency in reality when some specific kinds of samples are highly sensitive. For the student model, 6 and 9 are mythical digits it has never seen. As listed in Table 1, without the teacher's knowledge, the student cannot recognize 6 and 9, getting accuracy of 0. RONA can significantly improve the student's accuracy on 6 and 9 with a reasonable privacy loss even though the student has never seen 6 and 9 during training.

Table 2: Compression Performance

		# Params	Time(s)	Acc(%)
MNIST	T	155.21K	0.76	99.48
	S1	4.97K	0.03	98.94
	S2	9.88K	0.07	99.28
SVHN	T	1.41M	7.34	96.36
	S1	0.04M	0.29	94.49
	S2	0.07M	0.39	95.39
CIFAR-10	T	3.12M	13.92	86.35
	S1	0.15M	0.93	82.14
	S2	0.52M	3.10	84.57

Compression Performance

We randomly choose 80% training samples as the public data to validate the compression performance on the three datasets. For MNIST, SVHN, and CIFAR-10, we enforce $(9.60, 10^{-5})$, $(9.83, 10^{-6})$, and $(9.59, 10^{-5})$ differential privacy, respectively. In order to examine the runtime on mobile devices, we deploy these DNNs on HUAWEI HONOR 8 equipped with ARM Cortex-A53@2.3GHz and Cortex-A53@1.81GHz to process 100 images consecutively.

The results listed in Table 2 show that the models with larger sizes achieve better performance. On all the three datasets, the student models trained by RONA obtain comparable accuracies to the teacher models in spite of using much less capacity and fewer training data. On MNIST, the student achieves 15× compression ratio and 11 × speed-up with only 0.2% accuracy decrease. On SVHN, the student model obtains slightly worse result ($< 1\%$) than the teacher model, while requiring 20 times fewer parameters and 19 times less runtime. On CIFAR-10, the accuracy decreases

less than 2% while the model size is 6 times smaller. The above results argue that RONA can privately compress large models with acceptable accuracy loss.

Acknowledgments

This work is supported by the Scientific Research Project of National University of Defense Technology through grant ZK17-03-48, Science Fund for Distinguished Young Scholars in Hunan Province through grant 2018JJ1032, NSFC through grants 61872378, 61572511, 91648204, and 71702186, as well as NSF through grants IIS-1526499, IIS-1763325, and CNS-1626432.

References

Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H. B.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, 308–318.

Beimel, A.; Brenner, H.; Kasiviswanathan, S. P.; and Nissim, K. 2014. Bounds on the sample complexity for private learning and private data release. *Machine Learning* 94(3):401–437.

Bucilua, C.; Caruana, R.; and Niculescu-Mizil, A. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, 535–541.

Cao, B.; Zheng, L.; Zhang, C.; Yu, P. S.; Piscitello, A.; Zulueta, J.; Ajilore, O.; Ryan, K.; and Leow, A. D. 2017. Deepmood: modeling mobile phone typing dynamics for mood detection. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, 747–755.

Chen, G.; Choi, W.; Yu, X.; Han, T.; and Chandraker, M. 2017. Learning efficient object detection models with knowledge distillation. In *Advances in Neural Information Processing Systems 30*, NIPS '17. 742–751.

Claerhout, B., and DeMoor, G. 2005. Privacy protection for clinical and genomic data: The use of privacy-enhancing techniques in medicine. *International Journal of Medical Informatics* 74(2):257 – 265.

Dwork, C., and Roth, A. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* 9(3-4):211–407.

Dwork, C. 2011. *Differential Privacy*. Boston, MA: Springer US. 338–340.

Han, S.; Mao, H.; and Dally, W. J. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations*, ICLR '16.

Hinton, G.; Vinyals, O.; and Dean, J. 2014. Distilling the knowledge in a neural network. In *Advances in Neural Information Processing Systems Deep Learning Workshop*, NIPS '14.

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:1704.04861*.

Korupolu, M. R.; Plaxton, C. G.; and Rajaraman, R. 1998. Analysis of a local search heuristic for facility location problems. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, 1–10.

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

Laine, S. 2017. Temporal ensembling for semi-supervised learning. In *5th International Conference on Learning Representations*, ICLR '17.

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

Lee, P. 2017. Technology, media and telecommunications predictions. *Delloitte Touche Tohmatsu Limited*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; and Ng, B. W. A. Y. 2011. Reading digits in natural images with unsupervised feature learning. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS '11. 1–9.

Osia, S. A.; Shamsabadi, A. S.; Taheri, A.; Rabiee, H. R.; Lane, N. D.; and Haddadi, H. 2017. A hybrid deep learning architecture for privacy-preserving mobile analytics. *arXiv:1703.02952*.

Papernot, N.; Abadi, M.; Erlingsson, U.; Goodfellow, I.; and Talwar, K. 2017. Semi-supervised knowledge transfer for deep learning from private training data. In *5th International Conference on Learning Representations*, ICLR '17.

Papernot, N.; Song, S.; Mironov, I.; Raghunathan, A.; Talwar, K.; and Erlingsson, U. 2018. Scalable private learning with pate. In *6th International Conference on Learning Representations*, ICLR '18.

Park, S.; Park, J.-K.; Shin, S.-J.; and Moon, I.-C. 2017. Adversarial dropout for supervised and semi-supervised learning. *arXiv:1707.03631*.

Romero, A.; Ballas, N.; Kahou, S. E.; Chassang, A.; Gatta, C.; and Bengio, Y. 2015. Fitnets: Hints for thin deep nets. In *3th International Conference on Learning Representations*, ICLR '15.

Sun, L.; Wei, X.; Zhang, J.; He, L.; Philip, S. Y.; and Srisaan, W. 2017. Contaminant removal for android malware detection systems. In *IEEE International Conference on Big Data (Big Data)*, 1053–1062.

Triastcyn, A., and Faltings, B. 2018. Generating artificial data for private deep learning. *arXiv:1803.03148*.

Wang, Z., and Ye, J. 2013. Querying discriminative and representative samples for batch mode active learning. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, 158–166.

Wang, K.; Zhang, D.; Li, Y.; Zhang, R.; and Lin, L. 2017. Cost-effective active learning for deep image classification.

IEEE Transactions on Circuits and Systems for Video Technology 27(12):2591–2600.

Wang, J.; Zhang, J.; Bao, W.; Zhu, X.; Cao, B.; and Yu, P. S. 2018. Not just privacy: Improving performance of private deep learning in mobile cloud. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, 2407–2416.