Privacy-Preserving Visual Learning Using Doubly Permuted Homomorphic Encryption (Supplementary Material)

Ryo Yonetani
The University of Tokyo
Tokyo, Japan
yonetani@iis.u-tokyo.ac.jp

Vishnu Naresh Boddeti
Michigan State University
East Lansing, MI, USA
vishnu@msu.edu

Kris M. Kitani
Carnegie Mellon University
Pittsburgh, PA, USA
kkitani@cs.cmu.edu

Yoichi Sato
The University of Tokyo
Tokyo, Japan
ysato@iis.u-tokyo.ac.jp

1. Some Statistics on Public/Private Images

We are interested in leveraging 'private' images, which are not shared publicly but just saved on a personal storage privately, for visual learning. In this section, we would like to provide some statistics that motivate our work.

Based on the recent report from Kleiner Perkins Caufield & Byers [14], the number of photos shared publicly on several social networking services (Snapchat, Instagram, WhatsApp, Facebook Messenger, and Facebook) per day has reached almost 3.5 billion in 2015. It also shows that the number of smartphone users in the world was about 2.5 billion in the same year. From these statistics, if everyone takes three photos per day on average, about four billion photos in total would be stored privately everyday. Some prior work [9, 10] has shown that such private photos still contained meaningful information including people, faces, and written texts, as well as some sensitive information like a computer screen and a bedroom. Our privacy-preserving framework is designed to learn visual classifiers by leveraging this vast amount of private images while preserving the privacy of the owners.

2. Examples of Privacy Leakage from Locally-Updated Classifiers

In our framework, users update classifier weights $\bar{\boldsymbol{w}}_t \in \mathbb{R}^D$ locally using their own private data and send the updated ones $\boldsymbol{w}_t^{(n)} \in \mathbb{R}^D$ to the aggregator. Here we discuss how the combination of $\bar{\boldsymbol{w}}_t$ and $\boldsymbol{w}_t^{(n)}$ can be used to reveal a part of the trained data.

Let us denote a labeled sample by $z_i = (\boldsymbol{x}_i, y_i)$ where $\boldsymbol{x}_i \in \mathbb{R}^D$ is a feature vector and $y_i \in \{-1, 1\}$ is a binary label. The whole data privately owned by a single user is then described by $\mathcal{Z} = \{z_i \mid i = 1, \ldots, K\}$. In order to learn a classifier, we minimize a regularized loss function which is defined with weights $\boldsymbol{w}$ and data $\mathcal{Z}$ as follows:

$$Q(\mathcal{Z}, \boldsymbol{w}) = \ell(\mathcal{Z}, \boldsymbol{w}) + \lambda R(\boldsymbol{w}), \tag{1}$$

where $\ell(\mathcal{Z}, \boldsymbol{w})$ is a loss function, $R(\boldsymbol{w})$ is a certain regularizer, and λ is a regularization strength.

2.1. Stochastic Gradient Descent

As described in Section 2.1 of the original paper, a part of trained private data could be leaked when users update a classifier via stochastic gradient descent (SGD). With SGD, users obtain weights $\boldsymbol{w}_t^{(n)}$ by updating $\bar{\boldsymbol{w}}_t$ based on the gradient of regularized loss with respect to single sample $z_t = (\boldsymbol{x}_t, y_t) \in \mathcal{Z}$ picked randomly from $\mathcal{Z}$ [3]:

$$\boldsymbol{w}_t^{(n)} = \bar{\boldsymbol{w}}_t - \gamma_t \nabla_{\bar{\boldsymbol{w}}_t} Q(z_t, \bar{\boldsymbol{w}}_t), \tag{2}$$

where γ_t is a learning rate at time step t and controls how much one can learn from the sample z_t. Loss gradient $\nabla_{\bar{\boldsymbol{w}}_t} Q(z_t, \bar{\boldsymbol{w}}_t)$ is described as follows:

$$\nabla_{\bar{\boldsymbol{w}}_t} Q(z_t, \bar{\boldsymbol{w}}_t) = \nabla_{\bar{\boldsymbol{w}}_t} \ell(z_t, \bar{\boldsymbol{w}}_t) + \lambda \nabla_{\bar{\boldsymbol{w}}_t} R(\bar{\boldsymbol{w}}_t). \tag{3}$$

By plugging Equation (3) into Equation (2), we obtain:

$$\nabla_{\bar{\boldsymbol{w}}_t} \ell(z_t, \bar{\boldsymbol{w}}_t) = \frac{\bar{\boldsymbol{w}}_t - \boldsymbol{w}_{t+1}^{(n)}}{\gamma_t} - \lambda \nabla_{\bar{\boldsymbol{w}}_t} R(\bar{\boldsymbol{w}}_t). \tag{4}$$

Now we are interested in what one can know about $z_t = (\bar{\boldsymbol{w}}_t, y_t)$ from Equation (4) *where both* $\bar{\boldsymbol{w}}_t$ *and* $\boldsymbol{w}_{t+1}^{(n)}$

are given. If the type of regularizer $R(\cdot)$ can be identified (*e.g.*, L2 regularization), $\nabla_{\bar{\boldsymbol{w}}_t} R(\bar{\boldsymbol{w}}_t)$ can be computed exactly. In addition, if concrete parameters for γ_t and λ can be guessed (*e.g.*, when using a default parameter of open source libraries) and if a specific loss function is used for $\ell(z_t, \bar{\boldsymbol{w}}_t)$, one can narrow down the private sample z_t to several candidates.

Specifically, let Θ be the RHS of Equation (4), which is given when $R(\cdot)$ is identified and γ_t, λ is estimated. Then, when using some specific loss functions, we can solve $\nabla_{\bar{\boldsymbol{w}}_t} \ell(z_t, \bar{\boldsymbol{w}}_t) = \Theta$ for z_t as follows:

Hinge loss $\nabla_{\bar{\boldsymbol{w}}_t} \ell(z_t, \bar{\boldsymbol{w}}_t) = \mathbf{0}_D$ (an all-zero vector of size D) if $1 - y_t \bar{\boldsymbol{w}}_t^\top \boldsymbol{x}_t < 0$ or $-y_t \boldsymbol{x}_t$ otherwise. If Θ is a non-zero vector, then $z_t = (\Theta, -1)$ or $(-\Theta, 1)$.

Logistic loss $\nabla_{\bar{\boldsymbol{w}}_t} \ell(z_t, \bar{\boldsymbol{w}}_t) = \frac{-y_t \boldsymbol{x}_t}{1+\exp(y_t \bar{\boldsymbol{w}}_t^\top \boldsymbol{x}_t)} = \frac{-X}{1+\exp(\bar{\boldsymbol{w}}_t^\top X)}$, where $\bar{\boldsymbol{w}}_t$ is known and $X = y_t \boldsymbol{x}_t$. X can be obtained numerically (*e.g.*, via the Newton's method), and $z_t = (X, 1)$ or $(-X, -1)$.

Note that this problem of sample leakage can happen also when users have just a single image in their storage.

2.2. Gradient Descent

When users have more than one image, it might be natural to use a gradient descent (GD) technique instead of SGD. Namely, we evaluate the loss gradient averaged over the whole data $\nabla_{\bar{\boldsymbol{w}}_t} \ell(\mathcal{Z}, \bar{\boldsymbol{w}}_t) = \frac{1}{K} \sum_i \nabla_{\bar{\boldsymbol{w}}_t} \ell(z_i, \bar{\boldsymbol{w}}_t)$ instead of that of a single sample. Although this averaging can prevent one from identifying individual sample z_i, the equation $\nabla_{\bar{\boldsymbol{w}}_t} \ell(\mathcal{Z}, \bar{\boldsymbol{w}}_t) = \Theta$ still reveals some statistics of private data $\mathcal{Z}$. Specifically, if the class balance of data is extremely biased, one can guess an average of samples that were not classified correctly. One typical case of class unbalance arises when learning a detector of abnormal events. This will regard most of training samples as negative ones.

Let us consider an extreme case where all of the samples owned by a single user belong to the negative class, *i.e.*, $y_i = -1 \ \forall z_i = (\boldsymbol{x}_i, y_i) \in \mathcal{Z}$. If we use the hinge loss, a set of samples that were not classified perfectly is described by $\bar{\mathcal{Z}} = \{z_i = (\boldsymbol{x}_i, y_i) \mid y_i \bar{\boldsymbol{w}}_t^\top \boldsymbol{x}_i < 1\} \subseteq \mathcal{Z}$. Then, the averaged loss gradient is transformed as follows:

$$\nabla_{\bar{\boldsymbol{w}}_t} \ell(\mathcal{Z}, \bar{\boldsymbol{w}}_t) = \frac{1}{K} \sum_{z_i \in \bar{\mathcal{Z}}} -y_i \boldsymbol{x}_i = \frac{1}{K} \sum_{z_i \in \bar{\mathcal{Z}}} \boldsymbol{x}_i = \Theta. \quad (5)$$

Namely, Θ is proportional to the average of samples that were not classified correctly.

For the logistic loss, when $y_i = -1$, the loss gradient with respect to a single sample becomes $\nabla_{\bar{\boldsymbol{w}}_t} \ell(z_i, \bar{\boldsymbol{w}}_t) = \frac{\boldsymbol{x}_i}{1+\exp(-\bar{\boldsymbol{w}}_t^\top \boldsymbol{x}_i)} = P(y_i = 1 \mid \boldsymbol{x}_i) \boldsymbol{x}_i$, where $P(y_i = 1 \mid \boldsymbol{x}_i)$ is close to 0 when z_i is classified correctly as negative, and increases up to 1 when classified incorrectly as positive. Then, the averaged loss gradient becomes:

$$\nabla_{\bar{\boldsymbol{w}}_t} \ell(\mathcal{Z}, \bar{\boldsymbol{w}}_t) = \frac{1}{K} \sum_i P(y_i = 1 \mid \boldsymbol{x}_i) \boldsymbol{x}_i = \Theta. \quad (6)$$

If all the samples are classified confidently, *i.e.*, $|\bar{\boldsymbol{w}}_t^\top \boldsymbol{x}_i| \gg 0 \ \forall z_i \in \mathcal{Z}$, Θ is again proportional to the average of samples not classified correctly.

Figure 1. **Image Reconstruction from Features** with [4]

2.3. Reconstructing Images from Features

The previous sections demonstrated that classifiers updated locally via SGD/GD could expose a part of trained feature vectors. We argue that users will further suffer from a higher privacy risk when the features could be inverted to original images (*e.g.*, [4, 13, 21]). Figure 1 showed examples on image reconstruction from features using [4] on some images from Caltech101 [6]. We extracted outputs of the fc6 layer of the Caffe reference network used in the open source library[1]. Although reconstructed images do not currently describe the fine-details of original images (*e.g.*, contents displayed on the laptop), they could still capture the whole picture indicating what were recorded or where they were recorded.

3. Additional Experimental Results

In the original paper, we evaluated our approach on a variety of tasks not only object classification on the classic Caltech Datasets [6, 7] but also face attribute recognition and sensitive place detection on a large-scale dataset [5, 12]. This section introduces some additional experimental results using different tasks or datasets.

Video Attribute Recognition on YouTube8M Subset We evaluated the proposed method (**DPHE**) as well as the two privacy-preserving baselines (**PPR10** [17], **RA12** [18]) on a video attribute recognition task using a part of YouTube8M dataset [1]. Specifically, we picked 227,476 videos from the training set and 79,398 videos from the validation set. Similar to the data preparation in Section 3.2 of the original paper, 50,000 videos of our training set were left for the initialization data and the rest was split into five

[1] http://lmb.informatik.uni-freiburg.de/resources/software.php

Table 1. **Video Attribute Recognition Results on the Part of YouTube8M Dataset**: mean average precision (mAP) for the top 10, 50, and 100 frequently-annotated attributes.

Methods	mAP (∼10)	(∼50)	(∼100)	Privacy
PRR10 [17]	0.62	0.45	0.38	✓
RA12 [18]	0.60	0.45	0.37	✓
No-PP	0.70	0.55	0.48	✗
DPHE	0.70	0.53	0.47	✓

to serve as private data with $N = 5$. Although over 4,000 attributes like 'Games,' 'Vehicle,' and 'Pina Records.' were originally annotated to each video, our evaluation used the top 100 frequent attributes that were annotated to more than 1,000 videos in our training set. For each video, we extracted outputs of the global average pooling layer of the deep residual network [8] trained on ImageNet [19] every 30 frame (about once in a second) and average them to get a single 2048-dimensional feature vector. Table 1 describes the mean average precision (mAP) for the top 10, 50, and 100 frequently-annotated attributes. We confirmed that DPHE outperformed the two privacy-preserving baselines. The sparsity of locally-updated classifiers was 65% on average, which resulted in about 3.5 minutes for the encryption. In order to see the original performance obtained by using residual network features, we introduced another baseline (**No-PP** in the table) that learned an L2-regularized linear SVM on the whole training data via SGD. The results demonstrated that the performance with DPHE was almost comparable to that with No-PP.

Clustering on Caltech101/256 Unlike the other privacy-preserving baselines [17, 18], DPHE can also be applied to an unsupervised clustering task based on mini-batch k-means [20]. Instead of learning a classifier with the initialization data, an aggregator first runs k-means++ [2] to distribute cluster centroids to users. Users then update the centroids locally with sparse constraints. We used the L1-regularized stochastic k-means [11] to obtain the sparse centroids. Regularization strength (η in [11]) was chosen adaptively so that the sparsity of cluster centroids was more than 90% on average. Table 2 shows an adjusted mutual information score of the clustering task on Caltech101 [6] and Caltech256 [7]. As a baseline method, we chose a standard mini-batch k-means (S10 [20]). For all of the methods, the number of image categories was given as the number of clusters, *i.e.*, we assumed that the correct cluster number was known. We found that DPHE achieved a comparable performance to the baseline method.

4. Security Evaluation

Finally, we introduce a formal version of our security evaluation on Algorithm 1 that supplements Section 2.5 in the original paper. Recall that our framework involves the following three types of parties:

Table 2. **Clustering Results on Caltech101/256**: adjusted mutual information scores given the correct number of clusters.

Methods	Caltech101	Caltech256	Privacy
S10 [20]	0.733	0.630	✗
DPHE	0.753	0.614	✓

Definition 1 (Types of parties)**.** *Let $U^{(1)}, \ldots, U^{(N)}$ be N users, A be an aggregator, and G be a key generator. We assume that they are all semi-honest [15] and do not collude. $U^{(n)}$ has private data $\boldsymbol{w}^{(n)} \in \mathbb{R}^D$. $U^{(n)}$ can communicate only with A and G, while A and G can communicate with all of the parties. $U^{(j)}$ $(j \neq n)$ may be malicious and intercept data sent from $U^{(n)}$ to A.*

With DPHE, private data $\boldsymbol{w}^{(n)}$ is first decomposed into $\boldsymbol{w}^{(n)} = K^{(n)}\boldsymbol{v}^{(n)}$, where $\boldsymbol{v}^{(n)} \in \mathbb{R}^M$, $K^{(n)} \in \{0,1\}^{D,M}$, and M is an encryption capacity indicating the maximum number of values that are Paillier encrypted. $K^{(n)}$ is called an index matrix and defined as follows:

Definition 2 (Index matrix)**.** *An index matrix of $\boldsymbol{w} \in \mathbb{R}^D$ given $\boldsymbol{v} \in \mathbb{R}^M$ is a binary matrix $K \in \{0,1\}^{D \times M}$ such that the number of ones for each column is exact one and that for each row is at most one and $\boldsymbol{w} = K\boldsymbol{v}$. Let $\mathcal{K}_{D,M}$ be a set of all possible index matrices of size $D \times M$.*

To make $\boldsymbol{v}^{(n)}$ secure, we use the Paillier encryption [16]. On the encrypted data $\boldsymbol{\zeta}\left(\boldsymbol{v}^{(n)}\right)$, the following lemma holds based on Theorem 14 and Theorem 15 in [16]:

Lemma 3 (Paillier Encryption [16])**.** *Let $\boldsymbol{\zeta}(\boldsymbol{v})$ be a vector which is obtained by encrypting $\boldsymbol{v}$ with the Paillier cryptosystem. Then, no one can identify $\boldsymbol{v}$ from $\boldsymbol{\zeta}(\boldsymbol{v})$ without a decryption key.*

Note that the Paillier encryption of $\boldsymbol{v}^{(n)}$ also helps to keep secret the number of non-zeros in $\boldsymbol{w}^{(n)}$ since M is always greater than or equal to the non-zero number.

On the other hand, DPHE doubly-permutes $K^{(n)}$, namely $\boldsymbol{\Phi}\left(K^{(n)}\right) = \boldsymbol{\phi}^{(n)}\boldsymbol{\phi}K^{(n)}$, where $\boldsymbol{\phi}^{(n)}, \boldsymbol{\phi} \in \{0,1\}^{D \times D}$ is a permutation matrix defined as follows:

Definition 4 (Permutation matrix)**.** *A permutation matrix that permutes D elements is a square binary matrix $\boldsymbol{\phi} \in \{0,1\}^{D \times D}$ such that the number of one for each column and for each row is exact one. Let Ω_D be a set of all possible permutation matrices of size D.*

In DPHE, $U^{(n)}$ has both $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$ but does not have $\{\boldsymbol{\phi}^{(j)} \mid j \neq n\}$, while A has $\{\boldsymbol{\phi}^{(n)} \mid n = 1, \ldots, N\}$ but does not have $\boldsymbol{\phi}$. In what follows we prove that without having both of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$, one cannot identify $K^{(n)}$ from

$\mathbf{\Phi}\left(K^{(n)}\right)$ (*i.e.*, only $U^{(n)}$ can identify $K^{(n)}$). As preliminaries, we introduce several properties of index and permutation matrices based on Definition 2 and Definition 4.

Corollary 5 (Properties of index matrices)**.** *For $\boldsymbol{\phi} \in \Omega_D$ and $K \in \mathcal{K}_{D,M}$, $\boldsymbol{\phi}K \in \mathcal{K}_{D,M}$.*

Corollary 6 (Properties of permutation matrices)**.** *For $\boldsymbol{\phi}, \boldsymbol{\phi}' \in \Omega_D$, $\boldsymbol{\phi}^{-1} = \boldsymbol{\phi}^\top \in \Omega_D$, and $\boldsymbol{\phi}\boldsymbol{\phi}' \in \Omega_D$.*

Then, the following lemma about $\mathbf{\Phi}\left(K^{(n)}\right)$ holds:

Lemma 7 (Reordering $\mathbf{\Phi}\left(K^{(n)}\right)$)**.** *Let $\mathbf{\Phi}\left(K^{(n)}\right) = \boldsymbol{\phi}^{(n)}\boldsymbol{\phi}K^{(n)} \in \mathcal{K}_{D,M}$ where $\boldsymbol{\phi}^{(n)}, \boldsymbol{\phi} \in \Omega_D$ and $K^{(n)} \in \mathcal{K}_{D,M}$. Suppose that $\mathbf{\Phi}\left(K^{(n)}\right)$ is known and $K^{(n)}$ is unknown. Then, $K^{(n)}$ can be determined uniquely if and only if both $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$ are known.*

Proof. If both of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$ are known, $K^{(n)}$ can be determined uniquely as $K^{(n)} = \boldsymbol{\phi}^\top(\boldsymbol{\phi}^{(n)})^\top\mathbf{\Phi}\left(K^{(n)}\right)$ where the variables in the RHS are all known. To prove the 'only-if' proposition, we introduce its contrapositive: 'if at least one of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$ is unknown, $K^{(n)}$ cannot be determined uniquely.' Let $\boldsymbol{\phi}' = \boldsymbol{\phi}^\top(\boldsymbol{\phi}^{(n)})^\top \in \Omega_D$ (which is also a permutation matrix as shown in Corollary 6) which is unknown when at least one of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^{(n)}$ is unknown. For arbitrary $\boldsymbol{\phi}'$, $K^{(n)} = \boldsymbol{\phi}'\mathbf{\Phi}\left(K^{(n)}\right)$ is always an index matrix as shown in Corollary 5. Therefore, $K^{(n)}$ cannot be determined uniquely as long as $\boldsymbol{\phi}'$ is not fixed. This means that the contrapositive is true, and Lemma 7 is proved. □

The combination of Lemma 3 and Lemma 7 proves that the aggregator A and user $U^{(j)}$ $(j \neq n)$ cannot identify $U^{(n)}$'s private data $\boldsymbol{w}^{(n)}$ and its non-zero indices from encrypted data $\boldsymbol{\zeta}\left(\boldsymbol{v}^{(n)}\right), \mathbf{\Phi}\left(K^{(n)}\right)$. The remaining concern is if one can identify $\boldsymbol{w}^{(n)}$ or its non-zero indices from $\bar{\boldsymbol{w}} = \frac{1}{N}\sum_n \boldsymbol{w}^{(n)}$. As shown in the original paper, when $N \geq 3$, it is impossible for any party to decompose $\sum_n \boldsymbol{w}^{(n)}$ into individual $\boldsymbol{w}^{(n)}$'s or to decompose non-zero indices of $\bar{\boldsymbol{w}}$ into those of individual $\boldsymbol{w}^{(n)}$'s, as long as all parties are semi-honest and do not collude to share private information outside the algorithm.

To conclude, the following theorem is proved:

Theorem 8 (Security on Algorithm 1)**.** *After running Algorithm 1 by semi-honest and non-colluding parties $U^{(1)}, \ldots, U^{(N)}$, A, and G where $N \geq 3$, no one but $U^{(n)}$ can identify private data $\boldsymbol{w}^{(n)}$ and its non-zero indices from obtained information.*

References

[1] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. YouTube-8M: A Large-Scale Video Classification Benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 2

[2] D. Arthur and S. Vassilvitskii. K-Means++: the Advantages of Careful Seeding. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007. 3

[3] L. Bottou. Stochastic Gradient Tricks. In *Neural Networks, Tricks of the Trade, Reloaded*, pages 421–436. Springer, 2012. 1

[4] A. Dosovitskiy and T. Brox. Generating Images with Perceptual Similarity Metrics based on Deep Networks. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 1, pages 1–9, 2016. 2

[5] C. Fan and D. J. Crandall. Deepdiary: Automatically Captioning Lifelogging Image Streams. In *Proceedings of the Workshop on Egocentric Perception, Interaction, and Computing*, volume 9913, pages 459–473, 2016. 2

[6] L. Fei-Fei, R. Fergus, and P. Perona. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007. 2, 3

[7] G. Griffin, A. Holub, and P. Perona. Caltech-256 Object Category Dataset. Technical report, Caltech Technical Report 7694, 2007. 2, 3

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 171–180, 2016. 3

[9] R. Hoyle, R. Templeman, D. Anthony, D. Crandall, and A. Kapadia. Sensitive Lifelogs: A Privacy Analysis of Photos from Wearable Cameras. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1645–1648, 2015. 1

[10] R. Hoyle, R. Templeman, S. Armes, D. Anthony, and D. Crandall. Privacy Behaviors of Lifeloggers using Wearable Cameras. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous*, pages 571–582, 2014. 1

[11] V. Jumutc, R. Langone, and J. A. K. Suykens. Regularized and Sparse Stochastic K-Means for Distributed Large-Scale Clustering. In *Proceedings of the International Conference on Big Data*, pages 2535–2540, 2015. 3

[12] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015. 2

[13] A. Mahendran and A. Vedaldi. Understanding Deep Image Representations by Inverting Them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5188–5196, 2015. 2

[14] M. Meeker. Internet Trends 2016 – Code Conference, 2016. 1

[15] Oded Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004. 3

[16] P. Paillier. Public-Key Cryptosystems based on Composite Degree Residuosity Classes. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, volume 1592, pages 223–238, 1999. 3

[17] M. Pathak, S. Rane, and B. Raj. Multiparty Differential Privacy via Aggregation of Locally Trained Classifiers. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1876–1884, 2010. 2, 3

[18] A. Rajkumar and S. Agarwal. A Differentially Private Stochastic Gradient Descent Algorithm for Multiparty Classification. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 933–941, 2012. 2, 3

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 3

[20] D. Sculley. Web-Scale K-Means Clustering. In *Proceedings of the International Conference on World Wide Web*, pages 1177–1178, 2010. 3

[21] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. HOGgles: Visualizing Object Detection Features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–8, 2013. 2