A Hybrid Deep Learning Architecture for Privacy-Preserving Mobile Analytics

Seyed Ali Ossia*, Ali Shahin Shamsabadi*, Ali Taheri*, Hamid R. Rabiee*, Nic Lane‡•, Hamed Haddadi†

*Sharif University of Technology, ‡University College London, •Nokia Bell Labs, †Queen Mary University of London

ABSTRACT

The increasing quality of smartphone cameras and variety of photo editing applications, in addition to the rise in popularity of image-centric social media, have all led to a phenomenal growth in mobile-based photography. Advances in computer vision and machine learning techniques provide a large number of cloud-based services with the ability to provide content analysis, face recognition, and object detection facilities to third parties. These inferences and analytics might come with undesired privacy risks to the individuals.

In this paper, we address a fundamental challenge: Can we utilize the local processing capabilities of modern smartphones efficiently to provide desired features to approved analytics services, while protecting against undesired inference attacks and preserving privacy on the cloud? We propose a hybrid architecture for a distributed deep learning model between the smartphone and the cloud. We rely on the Siamese network and machine learning approaches for providing privacy based on defined privacy constraints. We also use transfer learning techniques to evaluate the proposed method. Using the latest deep learning models for Face Recognition, Emotion Detection, and Gender Classification techniques, we demonstrate the effectiveness of our technique in providing highly accurate classification results for the desired analytics, while proving strong privacy guarantees.

1. INTRODUCTION

The increasing availability of diverse applications and sensors on smartphones, in addition to their generous processing and storage capabilities, have made them an essential and inseparable part of our daily lives. Many of the applications utilize a range of data obtained from the camera or other sensors available on the phone. Majority of applications are free, relying on information harvesting from their users' personal data for targeted advertising. This practice has a number of privacy concerns and resource impacts for the users [26, 40]. Preserving individuals' privacy, versus detailed data analytics, face a dichotomy in this space. Cloud-based machine learning algorithms can provide beneficial or interesting services (e.g., health apps, or an image-based search app), however, their reliance on excessive data collection form the users can have consequences which are unknown to the user (e.g., face recognition for targeted social advertising). While complete data offloading to a cloud provider can have immediate or future potential privacy risks, techniques relying on performing complete analytics at the user end, or encryption-based methods, also come with their own resource limitations and user experience penalties (see Section 7 for detailed discussions).

In this paper, we focus on achieving a compromise between resource-hungry local analytics on a smartphone, versus privacy-invasive cloud-based services. We use smartphone as a system where tasks such as machine learning, specifically *Deep Learning* (DL), has been implemented successfully. We use image analytics (e.g., gender classification, emotion detection, and face recognition) as examples of desired tasks. Recently, deep learning have been used for many image analysis tasks [13, 39, 46]. More specifically, Convolutional Neural Network (CNN) is one of the powerful instances of deep models for doing image analysis. In order to achieve high accuracy in a specific task like object recognition, CNNs are trained using a large training dataset. More recently, a number of works [20,23,24] address the problem of inference using[1] deep models on mobile phones. Using complex and accurate deep models in mobile phones however requires significant processing and memory resources. On the other hand, cloud-based mobile computing models have a number of privacy risks [17, 37]. For example, when an application sends its user's picture to the cloud for image analysis tasks like tagging or annotating, sensitive information form the user or other individuals in the picture such a identity, age or race can be disclosed.

Apart from the resource considerations, an analytics service or an app provider might not be keen on sharing their valuable and highly tuned trained model. Hence, it is not always possible to assume local processing (e.g., a deep learning model on a smartphone)

[1]In this paper, by inference we mean applying a pre-trained deep model on an input to obtain the output, which is different from statistical inference.

is a viable solution even if the task duration, memory and processing requirements are not important for the user or tasks can be performed when the user is not actively using their phone (e.g., while the phone is being charged overnight). Some solutions for protecting data privacy have been centered around data reduction (e.g., via blurring and sampling), although most data reduction methods decrease accuracy for all tasks. Alternatively, noise addition to the source data has been suggested in recent works (see 7 for detailed discussions) to achieve privacy while allowing certain broader data analytics tasks to be carried on the data. As an exemplar use case for this paper, we consider a case where we wish to enable specific inference tasks such as gender classification or emotion detection on images taken on a smartphone, while protecting against a privacy-invasive task such as face recognition by a cloud operator having access to rich training data and pre-trained models (e.g., Google and Facebook). Here, we refer to the first solution (where all the inference takes place locally) and the second solution (where all the inference happens in the cloud) as the mobile and the cloud solutions, respectively.

In this paper, we design and evaluate a hybrid architecture where the mobile system and the cloud provider collaborate on completing the inference task. In this way, we can augment the mobile system to benefit from the cloud processing efficiency while addressing the privacy concerns. We concentrate on data mining applications where in order to get certain services from a provider, sending the data to the cloud is inevitable. As a specific exemplar of this general class of services, we consider image processing applications using deep learning. We address the challenge of performing certain approved image analytics in the cloud, without disclosing important information which could lead to other inferences such as identity leak via face recognition. Our approach relies on optimizing the layer separation of the pre-trained deep models. Primary layers are held on the mobile and the secondary ones on the cloud. In this way, the inference task starts by applying the primary layers of a feature extractor on the mobile, and continues by sending the resultant features to the cloud and, end by applying the secondary classification layers in cloud. We demonstrate that our proposed solution does not have the overhead of executing the whole deep model on the mobile, while it will be favored by a cloud provider as the user does not have access to their complete model and part of the inference should be done on the cloud. We introduce a method to manipulate the extracted features (from the primary layers) in a way that irrelevant extra information can not leak, hence addressing the privacy challenges of cloud solution. To do this, we alter the training phase by applying Siamese networks [10] and by employing a noise addition mechanism for increased privacy.

We evaluate our general approach using experiments on two applications with different deep models. The first one is gender classification and the second one is facial expression detection; in both cases, we want to do a task using cloud processing while simultaneously protecting the user identity against face recognition models in the cloud. In Section 3.3, we use two methods to quantify the privacy guarantees of our approach. One is to use *transfer learning* [47] which proves that face recognition is impractical by even using the state of the art models. Our other method provides us with strong guarantee by introducing a new way to quantify the privacy constraints. By strong guarantee, we mean that our method is model independent and prove that no learning model can compromise its privacy. Our applications and evaluations results in Section 4.2 show that our framework is general and can be used for any application which uses a deep learning model.

Our main contributions in this paper include:

- Proposing a machine learning framework for privacy-preserving mobile analytics on a cloud system and embedding deep networks on it;
- Developing a new technique for training deep models based on the Siamese architecture, which enables privacy at the point of offloading to the cloud;
- Performing evaluation of this framework across two common deep models: VGG-16 and VGG-S and two applications: gender classification and emotion detection in a way that we preserve the privacy relating to iface recognition.

2. MACHINE LEARNING FRAMEWORK

For enabling privacy-preserving analytics, we need a classification method with the following functions:

- Getting the input data (e.g. an image) from the client device (e.g., a smartphone);
- Performing the primary classification task (e.g. gender classification) on the client device;
- Offloading the resource-hungry machine learning operations to the cloud;
- Applying privacy-preserving measures to the data;
- Preserving the information needed for a secondary classification task (e.g. face recognition) from the server as much as possible.

In this part, we present a general framework for this objective. We define the main modules between the client device and the cloud server, and data communication between them. Figure 1 presents an overview of the framework. We can break down the analytics process into feature extraction and classification modules:

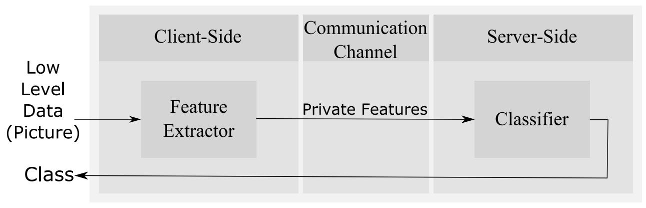


Figure 1: Privacy preserving machine learning framework

- *Feature Extractor*: This module gets the input data, operates an algorithm on the input data and outputs a new feature vector. This intermediate feature vector needs to keep the necessary information about the first classification task (CT_1), while protecting against the second classification task (CT_2) as much as possible. Usually, these two objectives are contradictory, i.e., decreasing the information available to CT_2 causes a decrease in the information available to CT_1 too. An ideal feature extractor module would keep enough information about CT_1 despite hiding information available to CT_2 as much as possible. The first objective could be quantified by evaluating the CT_1 classifier accuracy. The measure for the privacy-preservation will be explored in section 3.3.
- *Classifier*: This module gets the intermediate features, generated by the feature extractor, as its input for the CT_1 classifier. In practice, this module can be any ordinary classifier and privacy of intermediate data will be ensured by the first module (feature extractor).

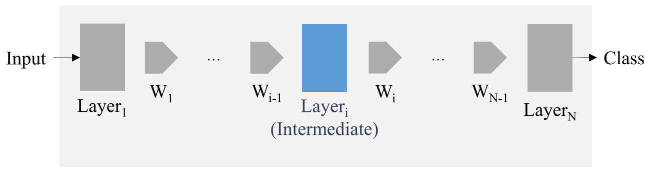
As most cloud providers do not set the user privacy as their primary concern, a validation method is needed for the user to ensure that their privacy is warranted. This validation method could be tailored based on the design of each module, so every instance of this framework needs a specific validation method. In order to use this framework in a specific scenario we should determine the followings:

- Choosing an appropriate CT_1 classifier.
- Designing a feature extractor and evaluate its privacy.
- Designing a privacy validation method for client

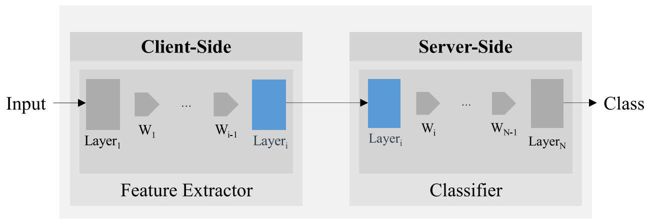
In Section 3, we explain our proposed system architecture based on this framework.

3. DEEP PRIV-EMBEDDING

Due to the increasing popularity of DL models in analytics applications, in this section we address how to



(a) Training simple embedding



(b) Using simple embedding. Intermediate layer is passed through communication channel.

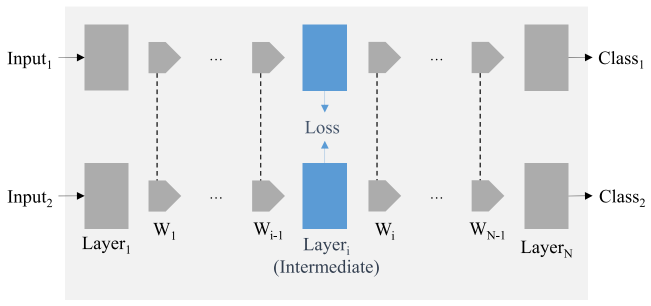
Figure 2: Simple embedding of a deep network

embed an existing DL model in our proposed framework. Complex deep networks consist of many layers and we use them in our framework using a layer separation mechanism. At the first step, we must choose the optimal intermediate layer from a deep network. Then we can store the layers before the intermediate layer on the mobile as a feature extractor, and the layers after that in the cloud server as the classifier (see Figure 1).

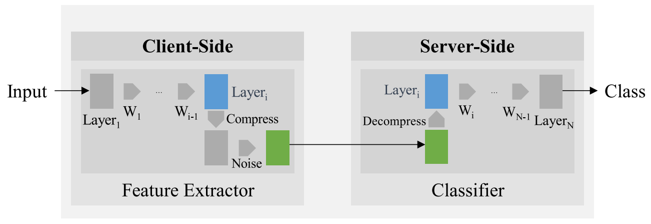
Choosing the intermediate layer from higher layers of the network intrinsically comes with privacy compromises. In [31], the authors reconstruct an original image from each layer and the accuracy of reconstruction decreases by using higher layers. As we go up through the deep network layers, the features get more specific to the classification task [47] and irrelevant information to the specific classification will be gradually lost. Hence, by using the layer separation mechanism, we achieve two important objectives simultaneously: (i) we end up with the feature extractor easily, and (ii) we benefit from the intrinsic characteristics of DL models for classification tasks. This approach satisfies the initial criteria we set for our proposed framework. In this paper, we refer to this embedding as the *simple embedding*. The training and test phase of this embedding can be seen in Figure 2. In section 6 we will evaluate the efficiency of this approach.

Moreover, experiments show that the accuracy of CT_1 does not decrease, when we reduce the dimension of the intermediate feature with Principle Component Analysis (PCA). Having done this, we can highly reduce the communication overhead between the client and server. We refer to this embedding (with PCA applied) as the *reduced simple embedding*.

Deep networks disentangle the underlying variations in training distribution [7]. They learn invariant fea-



(a) Training advanced embedding with Siamese structure where we have identical network structure and weights connected by dashed lines are equal.



(b) Using advanced embedding (with PCA projection and noise addition in client side and reconstruction and classification in server side)

Figure 3: Advanced embedding of a deep network

tures which are useful for many tasks at the same time. This useful feature of deep networks, learning general features, adversely affects the privacy of deep networks. For our exemplar task, we need to manipulate the intermediate layer and extract a new feature in a way that one could not identify the person in an image using a cloud-based face recognition model. One way to do this is to have a many to one mapping. This is the main idea behind k-*anonymity*. Suppose k different male images are mapped to one feature vector. Having this vector, an attacker will have confusion between k possible identities. We use the Siamese network [10] to accomplish this task. To the best of our knowledge, this is the first time that the Siamese network is used as a privacy preservation technique.

In order to increase the privacy when revealing mobile information to the server, we can fine-tune the existing deep model in a specific manner and test it in a different way. Our main contribution here relies on fine-tuning the model with the Siamese architecture, based on the chosen intermediate layer. Fine-tuning with Siamese architecture results in a feature space where objects with the same CT_1 classes cluster in together. Due to this transformation, classification borders of CT_2 get faded, consequently CT_2 becomes harder or even impossible, while the CT_1 is not affected. This approach makes the feature extractor more private, while preserving the privacy for the user against inference attacks on the cloud. We refer to this embedding as the *Siamese embedding*, where Siamese fine-tuning is applied. In addition, we can reduce the dimensions of the intermediate feature without any deficiency; we refer to this embedding method as the *reduced Siamese embedding.*

Another method which increases the client privacy and inference uncertainty of unauthorized tasks is noise addition. A service provider can determine a noise addition strategy for its clients in order to increase the uncertainty of other undesired tasks. We refer to *noisy embedding* whenever we use noise addition within the feature extractor. We refer to the noisy reduced Siamese embedding as the *advanced embedding.* In order to see the effect of Siamese fine-tuning, dimensionality reduction and noise addition, advanced embedding is shown in Figure 3.

When the Siamese network fine-tuning is applied, objects within the same class are clustered. By choosing the appropriate noise level, borders of all classification tasks, except CT_1, would be faded. This will not happen for simple embedding, because it does not have the property of Siamese feature space. As a result, the advanced embedding would be expected to have higher privacy protection against CT_2 than the noisy simple embedding, while it has almost the same accuracy of CT_1. Hence in the feature extractor module of advanced embedding, the following steps should be taken:

- Applying primary layers (which are fine-tuned with the Siamese network).
- Projecting the result on PCA Eigenvectors to reduce the dimensionality.
- Adding noise to the projection.

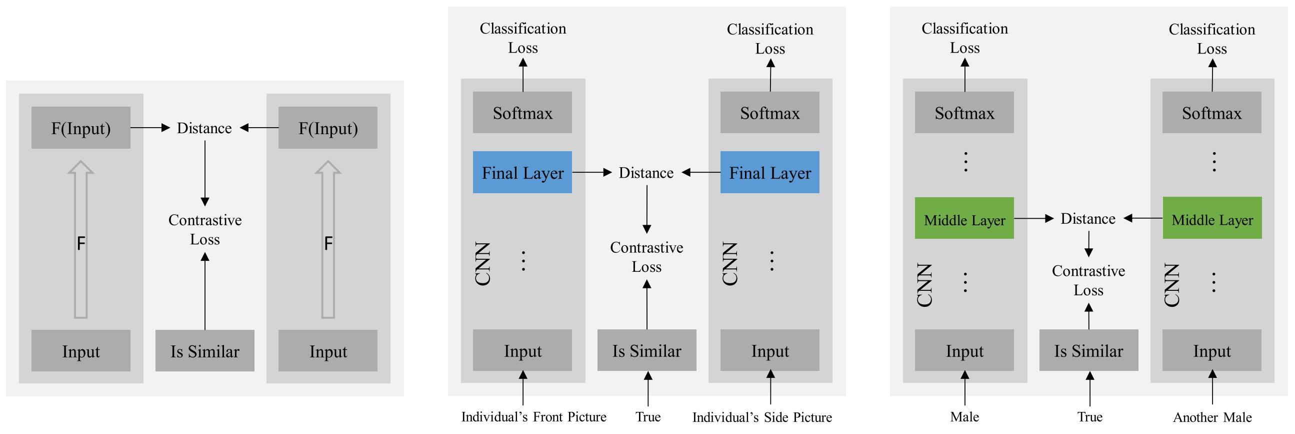
The classifier module should do these steps:

- Reconstructing the PCA projections.
- Applying remaining layers to get the final result.

In what follows we discuss our Siamese fine-tuning, dimensionality reduction and noise addition strategy in details.

3.1 Siamese Privacy Embedding

The Siamese architecture has been used in verification applications for long times [10]. It provides us with a kernel space, where similarity between the data points is defined in by their euclidean distance. The main idea of Siamese network is forcing the representations of two similar points to become near each other, and the representations of two dissimilar points become far. In order to do this, our training dataset should consists of pairs of similar and dissimilar points. For a pair of points, one function is applied to both of them and their value distance is computed. A contrastive loss function should



(a) Traditional Siamese architecture (b) Siamese architecture for classification (c) Siamese architecture for privacy

Figure 4: Siamese architecture usage

be defined in a way that making this distance maximize for two dissimilar points and being minimized for two similar points. An appropriate such loss function is defined in [16] and we use it in our application:

$$L(f_1, f_2) = \begin{cases} \|f_1 - f_2\|^2 & \text{similar} \\ \max(0, margin - \|f_1 - f_2\|)^2 & \text{dissimilar} \end{cases} \tag{1}$$

Where f_1 and f_2 are the mappings of data points. You can see a traditional architecture of Siamese network in figure 4a.

Consider the face verification application as an example. We want to determine whether two images belong to the same person or not. We should prepare a dataset consists of pairs of face images, some of them are similar and belong to the one person and some are not. Then by using a convolutional neural network as a feature extractor and imposing a contrastive loss function, we can train a similarity metric between face images.

Although the Siamese network is not designed for classification in its traditional definition, it works in our scenario. In [44], Sun *et al.* aim to create a face representation which is useful for both verification and identification tasks. They use a deep model which recognizes an identity with a softmax layer at the end. In this work, the last layer before the softmax provides a good representation for the image. Hence, pictures of the same person should be close to each other in this kernel space. They have used a Siamese network to force these representations to be close to each other. Having this constraint, they have a new Siamese network which also tries to classify each data points. You can see the architecture in Figure 4b. The two convolutional networks are the same and they are used just for training. In the test phase we just use one side of the network, so one of them is sufficient due to having the same parameters.

We take advantage of this kind of network to achieve our privacy preservation objectives. Our approach is based on the core idea of k-anonymity. Ideally we want to design a way to create a many to one mapping between objects with the same CT_1 class and different CT_2 class, (e.g. pictures of women with different personalities), to one point. In this case, CT_1 get more robust, while CT_2 (e.g. identification) get harder than before. Siamese network can provide us with this many to one mapping.

As an example, suppose CT_2 is an identification task. Unlike in [44] where the representations for the *same* person are forced to be near each other, we force mappings of *different* persons (objects with different CT_2 class) to be near each other. Our training data consists of pairs pictures from different identities. Defining similarity is based on CT_1. For the gender classification, we consider images in a pair as similar, if they are from the same gender (two men or two women), and dissimilar if they are not (one man and one woman). For this application, we use the Siamese network as in Figure 4c. Similarly for the emotion recognition, face images which have the same emotion are considered as similar and if they do not have the same emotion they are dissimilar.

Choosing the intermediate layer is another issue which we consider. We could use the previous layer of softmax as in [44], or we can use any other middle layer and define the contrastive loss on that. It is the service provider that decides how much of the network should be on mobile phone and how much of that should be in cloud. For example a service provider may decide to split the network from the fifth pooling layer. In this scenario, we should choose the output of this layer as our feature vector and define the contrastive loss on this layer. Sending the raw intermediate feature could burden the user with high communication cost. In or-

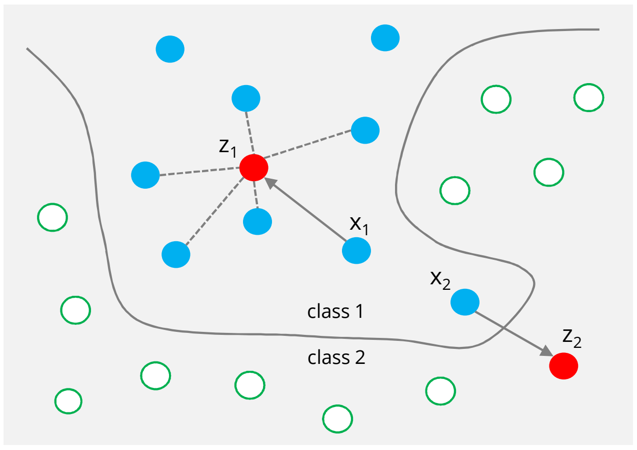


Figure 5: The effect of noise addition on the two dimensional intermediate feature space. Blue and green points show the objects of the first and second class, respectively. Consider x_1 and suppose we reach point z_1 after noise addition. Having z_1, we can not recognize the source node, because all the dotted lines have nearly the same distance and we are in doubt between these six points, which include 30% of all the points.

der to reduce this cost, we suggest using dimensionality reduction.

3.2 Dimensionality Reduction

After training with the Siamese structure, the service provider should collect intermediate features from all training data and apply PCA on them. The service provider should choose k as the dimension number of reduced space and give the projection matrix to the most informative k dimensional linear subspace (space of k principal components with higher Eigenvalues) to the mobile client. The user can use this projection matrix to create a highly reduced size feature and send it to the server. As the projection matrix is known to the service provider, it can reconstruct the feature from the reduced one, based on PCA eigenvectors. The most important benefit of this procedure is the reduction in communication cost. As we will show in Section 6, this procedure does not affect significantly on CT_1 accuracy.

3.3 Noise Addition

After reducing feature dimensionality, we add a Gaussian noise to each dimension of feature. The variance of this noise would be determined by the service provider based on their priorities. The effect of noising mechanism is shown in Figure 5. A Siamese network provides k-anonymity by mapping different objects from the same classes of CT_1 to the same point in feature space, while in practice, these points will have a small distance from each other, which is not negligible. Siamese network cluster objects from the same CT_1 classes and make them further from the others. Due to this closeness, the CT_1 class of a noisy point would remain fixed, while the CT_2 class of the noisy point would likely to be changed. After fine-tuning with the Siamese network, adding a small amount of Gaussian noise to each point results in a higher privacy of CT_2, without a significant decrease in CT_1 accuracy. Experiments in Section 6 confirm this conclusion.

This mechanism is not efficient in the simple embedding scenario in the absence of the Siamese feature space, where intra-class variance is decreased and inter-class variance is increased for CT_1. Experiments show that the Siamese fine tuning makes the tradeoff between CT_1 accuracy and CT_2 privacy significantly better. Naturally, increasing the noise variance causes an increase in the CT_2 privacy and a decrease in CT_1 accuracy. Hence we need a tradeoff between these two requirements based on their context by changing noise variance. The noise addition mechanism is efficient when we are in a low dimensional space. Comparison of distance is ineffective in high dimension and measuring uncertainty is more challenging. Consequently, we apply noise addition after reducing the dimensions of the intermediate features.

4. PRIVACY MEASUREMENT METHODS

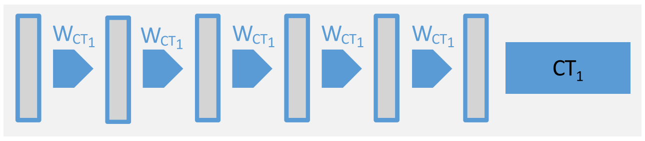
In this section we measure the privacy of extracted features for CT_1 in client side. We show the way in which information needed for CT_2 is removed and the extracted feature is specific to CT_1. In order to do this we use the transfer learning [47] approach for determining the degree of generality or specificity of features in each layer of deep networks

4.1 Transfer Learning

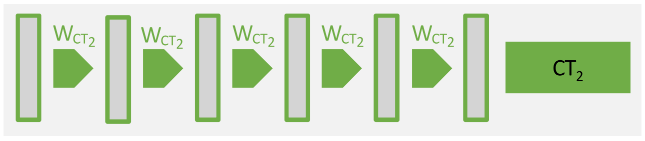
For determining the degree of generality or specificity of features in each layer of deep networks, we use transfer learning [47]. Suppose we have a trained network N_1 for CT_1 (Figure 6a). We build and train network N_2 for CT_2 (Figure 6b) with the following procedure:

- Copy weights from the first i layers of N_1 to the first i layers of N_2;
- Initialize the reminding layers of N_2 randomly (Figure 6c);
- Freeze the first i layers of N_2 (do not update their weights) ;
- Train N_2 on CT_2 (Figure 6d).

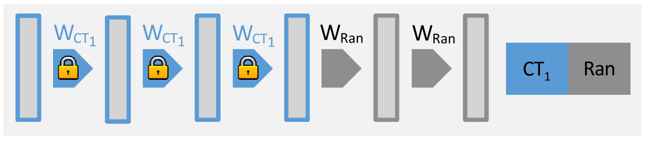
After the training procedure, the accuracy obtained for CT_2 is directly related to the degree of specificity or generality of the extracted feature from i'th layer. As we get lower general accuracy for CT_2, the feature is more specific to CT_1.



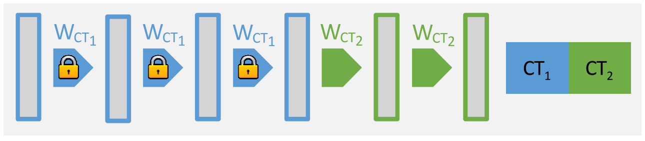
(a) Trained network for CT_1 (N_1)



(b) Network of CT_2 (N_2)



(c) Primary weight are copied from N_1 and frozen. The other layers have random weights.



(d) Trained network on CT_2 with transfer learning

Figure 6: Transfer Learning

4.2 Feature Extractor Validation

As the client part of the application is in the public domain, we require a method for confirming the feature extractor's privacy, based on some input samples. Specifically, the noise addition mechanism is necessary and evidently, should be known on the client side.

Suppose we have a dataset and we want to validate the privacy level of the feature extractor. We can get all the intermediate features and apply noise to them. Having all these features and a fixed noisy data point such as z, we can calculate the likelihood of each CT_2 class. In order to do this, we estimate $P(z|c_i)$ in this way:

$$\begin{aligned} P(z|c_i) &= \int_x P(z,x|c_i)dx \\ &= \int_x P(z|x,c_i)P(x|c_i)dx \end{aligned} \tag{2}$$

Conditioned on x, c_i is independent of z, so we have:

$$\begin{aligned} P(z|c_i) &= \int_x P(z|x)P(x|c_i)dx \\ &= E_{x\sim P(x|c_i)}[P(z|x)] \end{aligned} \tag{3}$$

Assuming $X_i = \{x_1, x_2, ..., x_{N_i}\}$ is the set of points from class c_i in our dataset, we can estimate the above expected value with sample mean; so we can estimate $P(z|c_i)$ with:

$$\widehat{P(z|c_i)} = \frac{1}{N_{c_i}} \sum_{\substack{x_j \in X \\ Class(x_j)=c_i}} P(z|x_j) \tag{4}$$

In this way, we can compute the relative likelihood of each class given a noisy data point. As we know the correct class of that point, we can determine the number of classes with a higher probability than the correct class. Hence, we can define the rank of likelihood of the right class as the privacy of that noisy point. We want this measure to have a normalized value between 0 and 1, so we divide it by T, the number of CT_2 classes:

$$Privacy(z) = \frac{Rank_{likelihood}(class(z))}{T}$$

Now, having intermediate features of N samples (with N noisy points generated by them), we can estimate the privacy of the transmitted data by:

$$Privacy_{\ total} = \sum_{i=1}^{N} \frac{Privacy(z_i)}{N}$$

We can define this as a measure for quantifying privacy. In the next sections, we simply refer to this metric as *Privacy*. With this measure, we can calculate how much privacy is preserved and also validate the privacy of the transmitted data.

5. DATASETS AND APPLICATIONS

In this section, we introduce the datasets used for our evaluations, and then we discuss the application scenarios. We apply transfer learning and privacy measurements to evaluate the efficiency of our proposed framework. We consider two important applications: gender classification and emotion detection, which have different deep structures. We consider the adversary classification task (CT_2) as face recognition. In order to evaluate with transfer learning, we use the state of the art VGG-16 model for face recognition, presented in [36]. We fine-tune their pre-trained model and get 75% accuracy on a 100 class classification task.

5.1 Datasets

We use the IMDB dataset for evaluating face recognition, Wiki and LFW dataset for gender classification and SFEW-2 for emotion detection.

5.1.1 IMDB

This dataset [36] contains 1,000 images from 2,622 highly-ranked celebrities on the IMDB website, for face recognition. The images are full height, so we detect and crop the faces for our recognition task. We randomly selected 100 different celebrities from this dataset

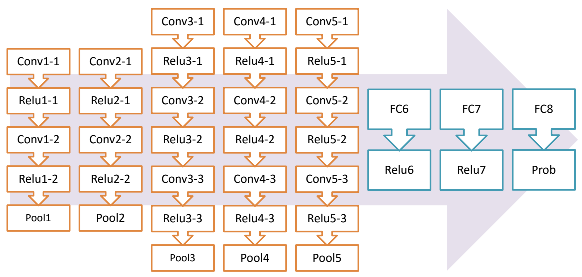


Figure 7: 16 layer VGG-16 structure [43]

and divide their images to training and validation sets to evaluate our face recognition model. For each person we use 500 images for training and 100 images for validation.

5.1.2 Wiki

Rothe *et al.* [41] prepared a huge dataset, named IMDB-Wiki, which is useful for age and gender estimation. We use the Wiki part of this dataset which contains 62,359 images to fine-tune our models. We use 45,000 images as training data and the rest as test data. We also evaluate our privacy measurement technique on this dataset.

5.1.3 LFW

Labeled Face in the Wild (LFW) [18] is an unconstrained face database containing 13,233 images from 5,749 individuals. It contains various possible daily conditions of people so the pose, lighting, race, accessories, occlusions, and background of its images are in natural settings. We use this dataset to test our gender classification models and compare them with others.

5.1.4 SFEW-2

Static Facial Expression in the Wild (SFEW) is an emotion detection benchmark [11]. We use the latest version [12] which consists of face images in seven emotional classes. This dataset contains 891 and 431 images for training and validation respectively.

5.2 Gender Classification

In a gender classification task, the goal is to classify an individuals' image to Male or Female. This has various applications in different systems such as human-computer interaction system, surveillance systems and targeted advertising [34]. Some techniques use face image as input to classifier, while others use the whole body image or a silhouette. In this paper we use cropped face images for the gender classification task. Recently, deep convolutional neural networks have been used for this problem [27,38,41]. In this work we use the model proposed in [41] with 94% accuracy, based on VGG-16, the popular 16-layer deep model for image classification [43] (see Figure 7).

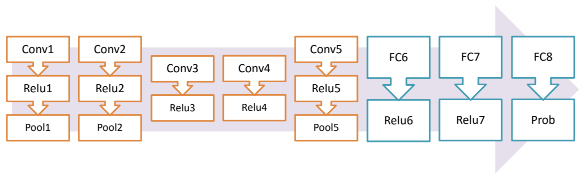


Figure 8: 8 layer VGG-S structure [9]

5.3 Emotion Detection

Emotion detection from facial expression is becoming exceedingly important for social media analysis tasks. In this problem, emotions are classified based on the individuals' facial expressions on images. Recently, deep learning has been demonstrated to be effective in solving this problem [28, 32]. Different deep models are proposed and compared in [28]. We choose the *VGG-S_RGB* model which is based on VGG-S structure [9], an 8-layer deep model which is popular for image classification (see Figure 8). The accuracy of doing emotion detection by using this model is 39.5% on SFEW-2 dataset.

6. EXPERIMENTS

In this section we evaluate and analyze the accuracy and privacy of different embeddings with different intermediate layers, by using our proposed measuring privacy tools: transfer learning, and privacy measurement. Although all of these embedding preserve privacy, applying Siamese fine-tuning is more efficient in a way that increase privacy considerably, whereas it does not decrease the accuracy of desired task. In addition, we show how dimensionality reduction has positive effects on privacy. Finally, we evaluate our framework on mobile phone and discuss its advantages regarding to other solutions.

6.1 Evaluation of Gender Classification

In this part, we apply transfer learning and privacy measurement on different intermediate layers of gender classification and face recognition models, in order to show the privacy of our framework. We set CT_1 as the gender classification task and CT_2 as the face recognition task. We use the VGG-16 model proposed at [41] in the simple embedding. In order to get the Siamese embedding, we use the pre-trained network of [41] and initiate a privacy Siamese structure (Figure 4c) with that. We divide the *Wiki* dataset to train and test set and fine tune our network based on a chosen intermediate layer. To create the reduced simple and Siamese embeddings, we apply PCA on the intermediate features of simple and Siamese embeddings, respectively. We choose 4, 6 and 8 as the PCA dimension for Conv5_3, Conv5_2 and

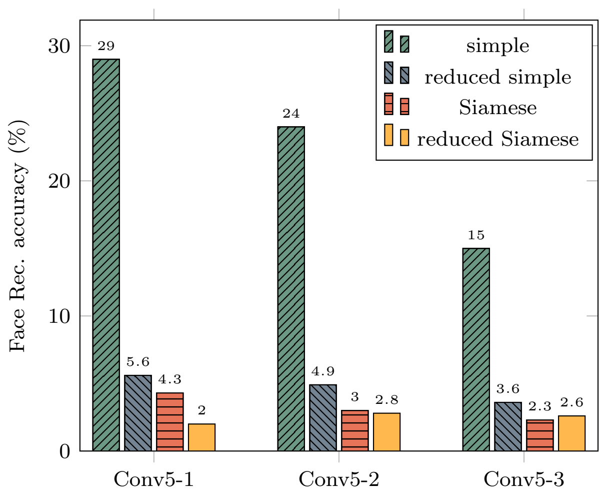


Figure 9: Gender Classification. Comparison of simple, reduced simple, Siamese and reduced Siamese embedding on different intermediate layers, while doing transfer learning. Accuracy of the original face recognition is 75%.

Conv5_1 respectively.

The result of transfer learning for different embeddings on different intermediate layers are presented in Figure 9. Overall, applying (reduced) simple or Siamese embedding results in a considerable decrease in the accuracy of face recognition from Conv5_1 to Conv5_3. The reason of this trend is that as we go up through the layers, the features of each layer will be more specific to the gender classification (CT_1). That is to say, the features of each layer do not have information related to face recognition (CT_2) as much as even its previous layer. In addition, for all of the layers, face recognition accuracy of Siamese embedding is by far less than the accuracy of simple embedding. This result has route in training of Siamese embedding with Siamese network which causes a dramatic drop in the accuracy. As it is shown in Figure 9, when Conv5_3 is chosen as the intermediate layer in Siamese embedding, the accuracy of face recognition is 2.3%, just ahead of random accuracy. Another interesting point of this figure is the effect of dimensionality reduction on the accuracy of face recognition. The reduced simple and Siamese embeddings has lower face recognition accuracy than simple and Siamese embedding, respectively.

To see how much these changes adversely affect accuracy of desired task which is gender classification, we report different embeddings accuracies in table 1. The results of table 1 convey two important messages. First, as the gender classification accuracy of Siamese and simple embedding are approximately the same, applying Siamese idea does not decrease accuracy of desired task. The other important result is that Siamese embedding is more robust to PCA than the simple embedding. In other words, gender classification accuracy of reduced Siamese embedding is close to Siamese embedding, whereas dimensionality reduction damage the accuracy of simple embedding. Figure 9 and table 1 show that applying the Siamese network and dimensionality reduction results in preserving privacy while gender classification accuracy does not decrease dramatically.

Table 1: Accuracy of Gender Classification on Different Embeddings. (PCA Dimension for reduced embeddings with Conv5-1, Conv5-2, and Conv5-3 as Intermediate Layer Is 8, 6, and 4 Respectively.)

Accuracy on LFW			
	Conv5-1	Conv5-2	Conv5-3
simple	94%	94%	94%
reduced simple	89.7%	87%	94%
Siamese	92.7%	92.7%	93.5%
reduced Siamese	91.3%	92.9%	93.3%

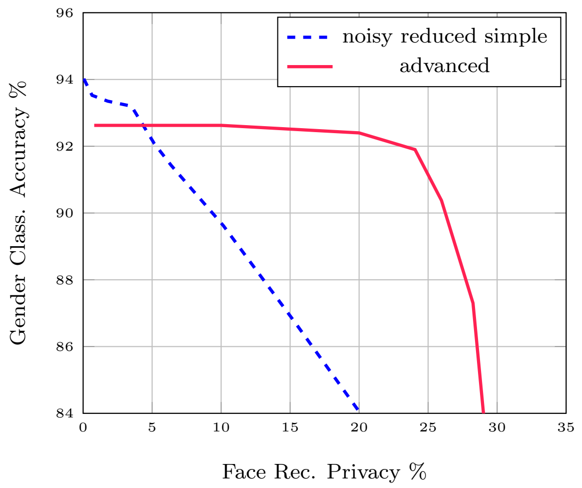


Figure 10: Accuracy vs. privacy for gender classification using VGG-16 structure in advanced embedding (conv5_3 is the intermediate layer)

In order to validate the feature extractor, we use the rank measure proposed in Section 4.2. By increasing the noise variance, we get more privacy and less accuracy. The service provider should gives us an accuracy-privacy curve (like Figure 10) and we can build exactly the same result with this kind of privacy measurement (which is independent of face recognition model).

In fact privacy and accuracy can be considered as two competing constraints and increasing the privacy of face recognition comes with a decrease in accuracy of gender classification. We show this dependency in Figure 10, where we can see the superiority of the advance embedding (noisy reduced Siamese) over noisy reduced simple embedding. From this figure, it is evident that by increasing privacy, gender classification accuracy de-

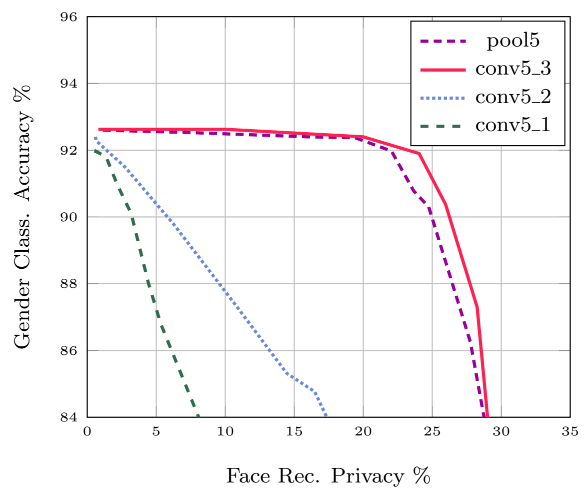


Figure 11: Accuracy vs. privacy for gender classification using VGG-16 structure and advanced embedding with different intermediate layers

creases more slowly in advanced embedding than other embeddings. This makes the advanced embedding the ideal choice as we have better privacy on a fixed accuracy level.

Another interesting experiment shows that choosing intermediate layers from higher ones, gives us better privacy for a fixed accuracy. This trend is shown in Figure 11, where the accuracy-privacy curve is upper for higher layers than lower layers. For a fixed accuracy, higher layer gives us more privacy. This validates our results of transfer learning in a way that choosing intermediate layers which are closer to the end of the network results in having a lower face recognition accuracy.

6.2 Evaluation of Emotion Detection

In this part, we want to evaluate our framework on emotion detection task, so CT_1 is emotion detection and CT_2 is face recognition. We use the VGG-S_RGB pretrained network of [28] in the simple embedding. We fine tune their model with Siamese structure on the training part of SFEW-2 dataset and get the Siamese embedding. As VGG-S has smaller structure in comparison with VGG-16 (8 layer vs. 16 layer), we just evaluate our embedding on one intermediate layer which is Conv5. We choose 10 as the PCA dimension and get reduced simple and Siamese embedding.

We test different embeddings with the transfer learning and the result are shown in Figure 12. The accuracy of the face recognition model is decreased for all embeddings. Similar to the gender classification application, the Siamese embedding works better than simple embedding and dimensionality reduction helps with privacy protection.

The effect of different embeddings on emotion detection are reported in Table 2. It is evident that the Siamese embedding does not decrease emotion detection accuracy significantly, while dimensionality reduction has amajor impact on this task.

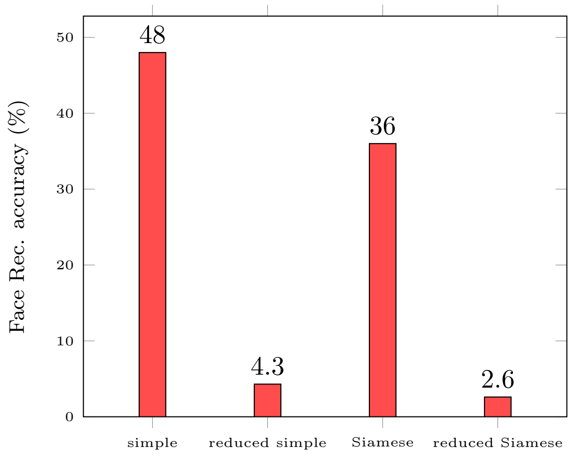


Figure 12: Emotion Detection. Comparison of simple, reduced simple, Siamese and reduced Siamese embedding on Conv5 as intermediate layer, while doing transfer learning. Accuracy of the original face recognition is 75%.

Table 2: Comparison of Different Deep Models Applied on Emotion Detection. Intermediate Layer is Conv5.

Accuracy on SFEW-2	
simple [28]	40%
Siamese	38%
reduced simple	31%
reduced Siamese	32%

The results of the feature extractor validation are shown in Figure 13, where the advanced embedding curve is above the noisy reduced simple curve. By having a fixed accuracy level, we can have higher privacy for advanced embedding. Results of the both applications show that our framework is application, and model, independent. The Siamese structure improves privacy, while reducing the dimensionality does not hurt the CT_1 accuracy and lowers the communication cost. We can use the validation method to quantify the privacy level, without access to the cloud-based face recognition model.

6.3 Mobile Evaluation

As shown in Figure 14 the number of parameters of the fully connected layers in the deep network are by far more than the number of parameters of the convolutional layers. In our framework, only some convolutional layers are loaded on the mobile, hence the memory overhead will see a significant drop, resulting in a considerable decrease in the model initialization

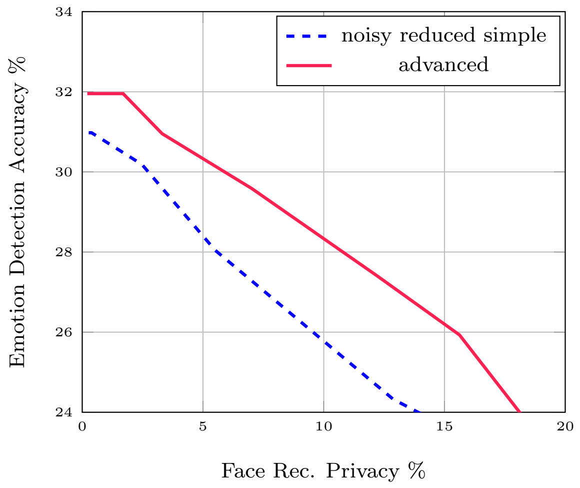


Figure 13: Accuracy vs. privacy for emotion detection using VGG-S structure and advanced embedding. (Conv5 is the intermediate layer)

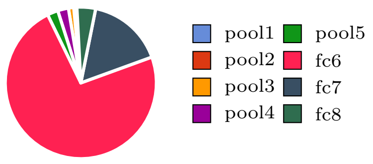


Figure 14: Percentage of parameters of each layer of VGG-S model

time and consequently power consumption [20]. More importantly, by reducing the dimensionality of the intermediate layer to less than 10, we only need to send a very low dimensional vector to the cloud. Although we perform some simple calculations on the mobile, this does reduce the communication cost considerably. Regarding the execution time, we utilize the cloud for the inference stage, so the only bottleneck is passing the input through the convolutional layers which are set on the mobile. Different methods such as compression [20] and sparsification [8] also try to reduce the cost of applying convolutional layers and can be used in our framework.

Most of the variations of trained model architectures under the proposed embedding approach have largely the same runtime performance. For the following experiments we use the same offloading layer as described earlier. Our smartphone implementation (based on Qualcomm Snapdragon 800) on average has an execution time and energy consumption per inference for the initial convolutional layers are executed locally as seen in Table 3. The values reported in this table are averages across all variations of architectures discussed but with offloading occurring (and the remainder of the model executed by the cloud) at the same layer in all. Table 3 does not report any of the cloud related energy or other resource costs.

Table 3: Average values over multiple runs for the Embedded Architecture

Latency	103msec
Energy	13mj

7. RELATED WORK

In this section, we describe the prior works on using deep learning on mobile phones and on privacy-preserving analytics systems.

7.1 Deep learning on mobile phone

Using pre-trained deep learning models can increase accuracy of different mobile sensors; e.g. in [23], Lane *et al.* use a 3 layer DNN which does not overburden the hardware. Complex networks with more layers need more processing power. DNN architectures such as the 16-layer model (*VGG-16*) proposed in [43] and the 8-layer model (*VGG-S*) proposed in [9] which are more complex, are implemented on the mobile in [20], and the resource usage such as time, CPU and energy overhead, are reported. As most of the state-of-the-art DNNs are pretty large in scale, fully evaluating all the layers on mobile results in serious drawbacks in processing time and memory requirements. Some methods are proposed to approximate these complex functions with simpler ones to reduce the cost of inference. Kim *et al.* [20] aim to compress deep models and in [8] the authors use sparsification and kernel separation. However, the increase in efficiency of these methods comes with a decrease in accuracy of the model.

There are several processor on the mobile which can be used for inferencing. CPU, GPU, and DSP are such processors. Alternative DNN models have been implemented which use the GPU for faster processing. The DNN implementation on GPU in [20] has burdens on the battery, hence it is not a feasible solution for some practical applications that either users frequently use it or continuously require it for long periods [22]. On the other hand, recent devices have DSP modules though their capacity for programming and storage can be limited. To tackle these problems, Lane *et al.* [22] have implemented a software accelerator called DeepX for large-scale DNN to reduce the resources while the mobile is doing inference by using different kinds of mobile processor simultaneously.

7.2 Learning with privacy

Prior works have approached the problem of privacy in machine learning from different point of views. Some approaches attempt to remove the irrelevant information by increasing the amount of uncertainty, while others try to hide information using cryptographic operations. Early works in this space mainly focus on publishing datasets for learning tasks [4, 5, 19, 45]. Their main concern is to publish a dataset consists of high level features for data mining tasks (e.g., medical database consisting of patients details), while preserving the individuals' privacy. Solutions such as randomized noise addition [4, 5], k-anonymity by generalization and suppression [25, 29, 30] are proposed and surveyed in [3]. These methods have some major problems. They are just appropriate for low-dimensional data due to the curse of dimensionality [2], hence they are not fit most of the multimedia data.A variety of attacks make many these methods unreliable [3].

Differential privacy [14] is another method provides an exact way to publish statistics of a database with specified amount of privacy. A learning model trained on some dataset can be considered as a high level statistic of that dataset. Recently [42] proposed concern of privacy for deep learning and [1] provided differential private deep learning model.

In dataset publishing, training applicability of a generalized data is important, while in this paper we deal with the cases where model training has been done already by a cloud service (e.g., Facebook or Google using their image data). In model publishing, mainly the privacy of users participating in training data is of concern. In our scenario, the user might want to protect a single image, hence neither publishing a dataset nor releasing statistics from it are directly relevant to our problem.

7.3 Privacy in image analytics

Privacy preservation has also been addressed in machine vision community. A good survey of all methods attempted to provide visual privacy, can be found in [35], which classifies different methods to five categories: intervention, blind vision, secure processing, redaction and data hiding. Our work is similar in spirit to de-identification works, a subcategory of redaction methods. The goal of these methods is to purturbe the individuals' faces in images in such a way that they can not be recognized by a face recognition system. A fundamental work in this category is presented in [33], which targets privacy issue in video surveillance data. The aim of this work is to publish a transformed dataset, where individuals are not identifiable. They show that using simple image filtering can not guarantee privacy and suggest *K-same* algorithm, based on k-anonymity, aiming to create *average* face images and replace them with the original ones. A shortcoming of this work is the lack of protection against future analyses on the dataset. Lots of works followed this idea and tried to improve it, mainly with the goal of publishing a dataset that is different from us. Their goal is not to protect privacy of a new face image, which is our concern. Follow-up works aim to transform a face image in a way that it is unrecognizable, while other analytics such as gender classification is possible. Most of the works in this area use visual filters or morphing to make the image unrecognizable [21, 38].

One of the main issues with prior privacy preservation methods is the lack of a privacy guarantee against new models due to engineering features against specific learning tasks. In most cases the learning task is not explicitly defined. Moreover, many works ignore the accuracy constraints of the learning task in their privacy preservation method. We address this issue in an adversarial setting. We optimize a cost function which consist of data privacy and model accuracy terms. We then use the Siamese network to solve this optimization.

An alternative approach to privacy preservation in machine learning is reliance on encryption. In [6], the authors provide a secure protocol for machine learning. In [15], the neural network is held in cloud. They encrypt the input of neural network in a way that inference becomes applicable on encrypted message. This approach has important, yet highly complex operations, making it infeasible for mobile systems. Mainly, the throughput is the same for inference on a single image or a batch. In addition neural network should be changed in a complex way to enable homomorphic encryption taking 250 seconds on a PC, which makes it impractical in terms of usability on a mobile.

8. DISCUSSIONS AND NEXT STEPS

In this paper, we presented a new hybrid framework for efficient privacy preserving analytics on mobile systems. Our framework consists of a feature extractor and classifier, where the former is placed on the client side and the later on the server side. We embed deep neural networks, specially convolutional neural networks in this framework to benefit from their accuracy and layered architecture. In order to protect the data privacy against unauthorized tasks, we used the Siamese architecture, creating a feature which is specific to the desired task. This is in contrast to today's ordinary deep networks in which the created features are generic and can be used for different tasks. Removing the undesired information from the extracted feature results in achieving privacy for the user. Evaluating our framework by splitting the layers between the mobile and the cloud and by targeted noise addition, we achieved high accuracy on the desired tasks, while heavily decreasing any inference potential for other tasks.

Our framework is currently designed for pre-trained machine learning inferences. In ongoing work we are extending our method by designing a framework for Learning as a Service, where the users could share their data, in a privacy-preserving manner, to train a new learning model. Another potential extension to our framework will be providing support for other kinds of neural networks such as recurrent neural network and also other applications for speech or video processing.

9. REFERENCES

[1] M. Abadi, A. Chu, I. Goodfellow, H. Brendan McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep Learning with Differential Privacy. *ArXiv e-prints*, July 2016.

[2] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st international conference on Very large data bases*, pages 901–909. VLDB Endowment, 2005.

[3] C. C. Aggarwal and S. Y. Philip. A general survey of privacy-preserving data mining models and algorithms. In *Privacy-preserving data mining*, pages 11–52. Springer, 2008.

[4] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 247–255. ACM, 2001.

[5] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *ACM Sigmod Record*, volume 29, pages 439–450. ACM, 2000.

[6] S. Avidan and M. Butman. Blind vision. In *European Conference on Computer Vision*, pages 1–13. Springer, 2006.

[7] Y. Bengio et al. Deep learning of representations for unsupervised and transfer learning. *ICML Unsupervised and Transfer Learning*, 27:17–36, 2012.

[8] S. Bhattacharya and N. D. Lane. Sparsification and separation of deep learning layers for constrained resource inference on wearables. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pages 176–189. ACM, 2016.

[9] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[10] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[11] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon. Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 2106–2112. IEEE, 2011.

[12] A. Dhall, O. Ramana Murthy, R. Goecke, J. Joshi, and T. Gedeon. Video and image based emotion recognition challenges in the wild: Emotiw 2015. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 423–426. ACM, 2015.

[13] P. N. Druzhkov and V. D. Kustikova. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26(1):9–15, 2016.

[14] C. Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[15] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 201–210, 2016.

[16] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[17] M. Haris, H. Haddadi, and P. Hui. Privacy leakage in mobile computing: Tools, methods, and characteristics. *arXiv preprint arXiv:1410.4978*, 2014.

[18] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[19] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 279–288. ACM, 2002.

[20] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530*, 2015.

[21] P. Korshunov and T. Ebrahimi. Using face morphing to protect privacy. In *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*, pages 208–213. IEEE, 2013.

[22] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, L. Jiao, L. Qendro, and F. Kawsar. Deepx: A software accelerator for low-power deep learning inference on mobile devices. In *2016 15th ACM/IEEE International Conference on*

Information Processing in Sensor Networks (IPSN), pages 1–12. IEEE, 2016.

[23] N. D. Lane and P. Georgiev. Can deep learning revolutionize mobile sensing? In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 117–122. ACM, 2015.

[24] N. D. Lane, P. Georgiev, C. Mascolo, and Y. Gao. Zoe: A cloud-less dialog-enabled continuous sensing wearable exploiting heterogeneous computation. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pages 273–286. ACM, 2015.

[25] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60. ACM, 2005.

[26] I. Leontiadis, C. Efstratiou, M. Picone, and C. Mascolo. Don't kill my ads!: balancing privacy in an ad-supported mobile application market. In *Proceedings of ACM HotMobile*, 2012.

[27] G. Levi and T. Hassner. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 34–42, 2015.

[28] G. Levi and T. Hassner. Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 503–510. ACM, 2015.

[29] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE, 2007.

[30] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[31] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *2015 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 5188–5196. IEEE, 2015.

[32] A. Mollahosseini, D. Chan, and M. H. Mahoor. Going deeper in facial expression recognition using deep neural networks. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10. IEEE, 2016.

[33] E. M. Newton, L. Sweeney, and B. Malin. Preserving privacy by de-identifying face images. *IEEE transactions on Knowledge and Data Engineering*, 17(2):232–243, 2005.

[34] C. B. Ng, Y. H. Tay, and B. M. Goi. Vision-based human gender recognition: A survey. *arXiv preprint arXiv:1204.1611*, 2012.

[35] J. R. Padilla-López, A. A. Chaaraoui, and F. Flórez-Revuelta. Visual privacy protection methods: A survey. *Expert Systems with Applications*, 42(9):4177–4195, 2015.

[36] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[37] L. Pournajaf, D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam. Participant privacy in mobile crowd sensing task management: A survey of methods and challenges. *ACM SIGMOD Record*, 44(4):23–34, 2016.

[38] N. Rachaud, G. Antipov, P. Korshunov, J.-L. Dugelay, T. Ebrahimi, and S.-A. Berrani. The impact of privacy protection filters on gender recognition. In *SPIE Optical Engineering+ Applications*, pages 959906–959906. International Society for Optics and Photonics, 2015.

[39] J. Rich, H. Haddadi, and T. M. Hospedales. Towards bottom-up analysis of social food. In *Proceedings of the 6th International Conference on Digital Health Conference*, DH '16, pages 111–120, New York, NY, USA, 2016. ACM.

[40] N. V. Rodriguez, J. Shah, A. Finamore, Y. Grunenberger, K. Papagiannaki, H. Haddadi, and J. Crowcroft. Breaking for commercials: characterizing mobile advertising. In *Proceedings of ACM Internet Measurement Conference*, pages 343–356, Nov. 2012.

[41] R. Rothe, R. Timofte, and L. Van Gool. Dex: Deep expectation of apparent age from a single image. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 10–15, 2015.

[42] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 1310–1321, New York, NY, USA, 2015. ACM.

[43] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[44] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2014.

[45] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty,*

Fuzziness and Knowledge-Based Systems, 10(05):557–570, 2002.

[46] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li. Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 157–166. ACM, 2014.

[47] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.