Mobile Sensor Data Anonymization

Mohammad Malekzadeh†⋄, Richard G. Clegg†, Andrea Cavallaro†, Hamed Haddadi⋄
†Queen Mary University of London, ⋄Imperial College London
m.malekzadeh, r.clegg, a.cavallaro@qmul.ac.uk   h.haddadi@imperial.ac.uk

ABSTRACT

Data from motion sensors such as accelerometers and gyroscopes embedded in our devices can reveal secondary undesired, private information about our activities. This information can be used for malicious purposes such as user identification by application developers. To address this problem, we propose a data transformation mechanism that enables a device to share data for specific applications (e.g. monitoring their daily activities) without revealing private user information (e.g. user identity). We formulate this anonymization process based on an information theoretic approach and propose a new multi-objective loss function for training convolutional auto-encoders (CAEs) to provide a practical approximation to our anonymization problem. This effective loss function forces the transformed data to minimize the information about the user's identity, as well as the data distortion to preserve application-specific utility. Our training process regulates the encoder to disregard user-identifiable patterns and tunes the decoder to shape the final output independently of users in the training set. Then, a trained CAE can be deployed on a user's mobile device to anonymize sensor data before sharing with an app, even for users who are not included in the training dataset. The results, on a dataset of 24 users for activity recognition, show a promising trade-off on transformed data between utility and privacy, with an accuracy for activity recognition over 92%, while reducing the chance of identifying a user to less than 7%.

CCS CONCEPTS

• Security and privacy; **• Human-centered computing → Ubiquitous and mobile computing**; **• Computing methodologies → Machine learning approaches**;

KEYWORDS

Sensor Data Privacy, Adversarial Training, Deep Learning, Time Series Analysis, Edge Computing.

1 INTRODUCTION

Motion data from sensors in user devices can reveal private information about users without their consent. For instance, motion patterns can reveal the user identity for user authentication [22]. However, other untrusted applications with direct access to raw sensory data may infer private information (e.g. inferring passwords from accelerometer data [24]). We are therefore interested in designing a privacy-preserving method that gives apps an on-device transformed version of the sensor data to prevent them to extract sensitive information unrelated to their task while preserving task-specific patterns (see Figure 1).

Differentially private mechanisms and information theoretic frameworks can be used to ensure privacy for data release. Although differential privacy [8, 34] offers a strong privacy guarantee for

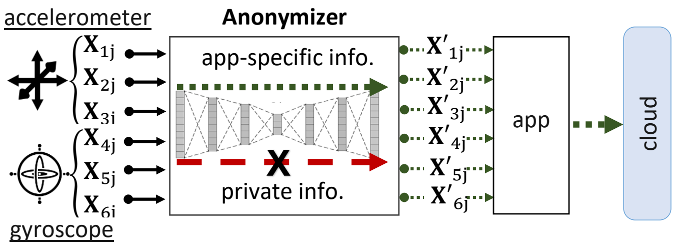

Figure 1: The Anonymizer (a pre-trained convolutional autoencoder) transforms raw sensory data before sharing with apps. An untrusted app can infer its required information, but it cannot infer user's private information. X_{ij} is the instant raw sensor data and X'_{ij} is the corresponding transformed value.

access to private datasets, it is not helpful for continuously releasing a user's sensor data. In fact, to design a private mechanism for publishing data, we need to aggregate all users' data in one place[1]. In our case, we do not trust a data aggregator and want to run the mechanism on the user's devices. Moreover, The local version of differential privacy [7, 16] is also very challenging for this setting. For the time series of sensor data, we repeatedly observe highly frequent patterns in consecutive data release, and applying the same differentially private mechanism to all of the released time window of data cannot provide a privacy guarantee, unless considerable noise is added to each window, that would eliminate the utility of the data in long-term data release [31].

Another approach to privacy-preserving data release is based on information theory [17, 28], where the mutual information, between the released data and the latent information which can be drawn from data, is considered as the measure of privacy. In this approach, we do not necessarily need to design a noise addition mechanism and it allows us to not only consider removing the private information but also care about keeping the useful information of the data [26]. To design a data release mechanism, that satisfies both utility and privacy constraints at the same time, we make a profit from adversarial training approaches [18]. Practically, we can approximate the mutual information by estimating the posterior distribution of the private variables given the released data [32], using the adversarial training mechanisms [9]. Therefore, information theoretic frameworks better fit our requirements of mobile sensor data release.

Existing solutions need a trusted party to have access to users' personal data to offer a reliable data distortion mechanism [16, 23, 35] or need to ask users to participate in a privacy-preserving training mechanism [1]. We are instead interested in anonymizing data locally and in defining a mechanism that can be shared across users.

[1]https://machinelearning.apple.com/2017/12/06/learning-with-privacy-at-scale.html

We formulate the problem as an optimization problem and propose a practical method to solve it. Our method is inspired by recent advances in adversarial training [18] for automatically discovering the useful representations needed for a specific task from raw data. We propose a new multi-objective loss function to train convolutional auto-encoders (CAEs) [21, 33]. This loss function regulates the transformed data to give as little information as possible about the user's identity; subject to minimum possible distortion in the data to preserve the utility. Unlike other approaches [12, 13, 28, 32] the training process not only regulates the encoder to only consider task-specific features included in the data but also tunes the decoder to shape the final output independent of the specific users in the training set. Therefore, we obtained a generalized model that can be applied to a new unseen user data, without user-specific re-training. Finally, we evaluate the efficiency and utility-privacy trade-off of the proposed mechanism, in comparison to other methods, on our own collected dataset of activity recognition[2] [19].

In summary, the major contributions of this paper are to formulate the problem of sensor data anonymization as an optimization process based on information theory and consequently to provide a new way of training deep auto-encoders by introducing a multi-objective loss function. We show how our multi-objective loss function approximates the desired solution to our optimization formulation in the specific case of activity and identity recognition. Importantly, we obtain a generalized model that can be applied to unseen data of new users, without the need to re-train the model.

2 RELATED WORK

Adversarial learning enables us to well approximate the underlying distribution of a data (e.g. with generative adversarial networks (GANs) [11]) or to model data with the well-known probability distributions (e.g. with variational auto-encoders (VAE) [14]). These techniques have been recently applied to calculate mutual information for solving optimization problems [9, 13, 23, 32]. The adversarial approach can be used to remove sensitive information from the latent low-dimensional representations of the data, e.g. by removing text from images [9].

An optimal privacy mechanism can be formulated as a game between two players, a privatizer and an adversary, with an iterative minimax algorithm [13]. Moreover, the service provider can share a feature extractor based on an initial training set that is then re-trained by the user on their data and then sent back to the service provider [23, 29]. However, in our work, we do not assume the existence of a trusted data aggregator and have access only to a public dataset for training. Moreover, the training cannot use the data of all the users.

Feature maps of a Convolutional Auto-Encoder (CAE) have the ability to extract patterns and dependencies among data points of a vector and have shown good performance in time series analysis [36]. CAE is usually trained by minimizing the differences (e.g. mean squared error or cross entropy) between the input and its reconstruction [21, 33]. They compress their input into a lower-dimensional latent representation and then reconstruct the input from this representation. The bottleneck of the CAE forces the optimization process to capture the most descriptive patterns in the data in order to generalize the model and prevent undesirable memorization [20, 21]. The information bottleneck [4, 10] in the middle layer of the CAE helps the encoding part of the CAE to capture the main factors of variation in the data into this latent representation. A simple way to train CAE is to randomly corrupt the input and force the model to refine it in the reconstruction [33].

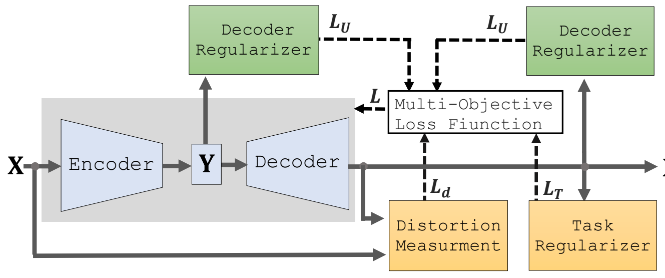

Figure 2: The adversarial training framework to train the CAE. After the training phase, we use only the CAE (Encoder and Decoder parts) to run on the users' device. Solid lines show data flow and dashed lines show loss functions.

With this approach, the CAE captures the prominent patterns and ignores the noise, so the reconstructed output does not longer contain meaningless patterns (i.e. noise). Moreover, a latent representation can be learned that removes some *meaningful* patterns from the data to reduce the possibility of inferring sensitive information [9, 16]. However, existing methods only consider the latent representation produced by the encoder part of CAE, and leave intact the decoder, which contains a large amount of information extracted from the data during training, thus limiting the protection.

3 SENSOR DATA ANONYMIZATION

3.1 The architecture

Let[3] sensor component [4] i, at sampling instant j, generate $\mathbf{X}_{\mathsf{ij}} \in \mathbb{R}$. Let the time series generated by M sensor components in a time window of length W, be represented by matrix $\mathbf{X} \in \mathbb{R}^{M \times W}$, with $\mathbf{X} = (\mathbf{X}_{\mathsf{ij}})$. Let N be the number of users. Let $\mathbf{U} \in \{0,1\}^N$ be a variable representing user identity. It is a one-hot vector of length N, a vector with 1 in the k-th place and 0 in all other places if user k generate the data $\mathbf{X}$ being considered. Let $\mathbf{T} \in \{0,1\}^B$ be a variable representing the current activity that generates $\mathbf{X}$. It is also a one-hot vector of length B.

Let us define the data with obscured user's identifiable information as the *anonymized sensor data*, $\mathbf{X}'$. We aim to produce anonymized sensor data so that the user's identity cannot be inferred by an untrusted app that has access to the sensor to recognize a set of B required activities. We use the concept of mutual information to consider how much can be inferred about a particular variable from a data set. We wish to minimize the amount the data

[2]https://www.kaggle.com/malekzadeh/motionsense-dataset

[3]As notation we use capital bold-face, e.g. $\mathbf{X}$, for random variables (univariate or multivariate) and lowercase bold-face, e.g. $\mathbf{x}$, for an instantiation; italic roman typestyle, e.g. I, for operations or functions; sans serif typestyle in lowercase, e.g. i, for indexing; and capital math font, e.g. M, for specific numbers such as the size of a vector.

[4]Like the z axis value of the gyroscope sensor.

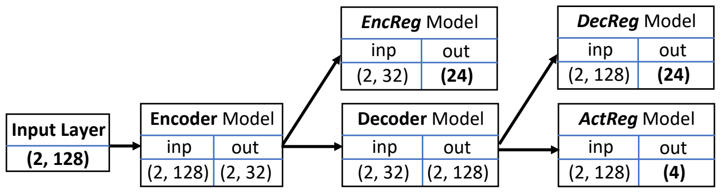

Figure 3: The overall architecture for the training procedure: the combined model in Figure (6).

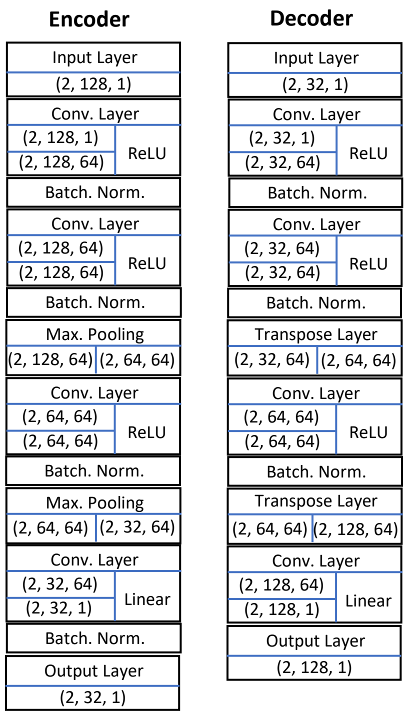

Figure 4: The CAE architecture: Encoder and Decoder.

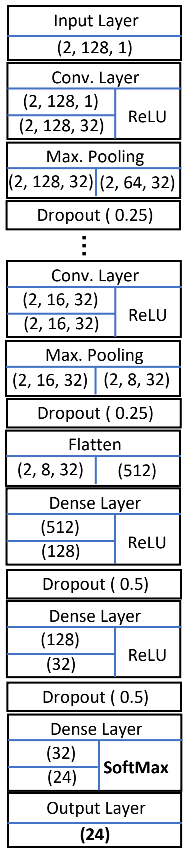

Figure 5: The *DecReg* architecture. Same structure for *EncReg* and *ActReg*.

changes but remove the ability to infer private information from the data.

We define $I(\cdot;\cdot)$ as the mutual information between two data items and $d(\cdot,\cdot)$ as some distance function between two data items (for example time series).

Definition 1. *Let $A(\mathbf{X})$ be some function applied to the data $\mathbf{X}$ that we consider for its potential to anonymize the data. We define the fitness function $F(A(\mathbf{X}))$ of $A(\mathbf{X})$ as*

$$F(A(\mathbf{X})) = \beta_U I\big(\mathbf{U}; A(\mathbf{X})\big) - \beta_T I\big(\mathbf{T}; A(\mathbf{X})\big) + \beta_d d\big(\mathbf{X}, A(\mathbf{X})\big), \quad (1)$$

*where β_U, β_T and β_d are variables that allow tuning the trade-off between concealing the user's identity, keeping required data and not distorting the data. We define the **anonymization function** $\mathcal{A}(\mathbf{X})$ as the function that best minimizes $F(A(\mathbf{X}))$ over all possible $\mathbf{X}$.*

Therefore, if $\mathbf{X}' = \mathcal{A}(\mathbf{X})$, then the optimal $\mathcal{A}(\cdot)$ transforms $\mathbf{X}$ into an $\mathbf{X}'$ such that $\mathbf{X}'$ will contain as little information as possible about the user's identity, $\mathbf{U}$, while maintaining activity distinctive information and minimizing the distortion of the original data. Note that, in practice, we find optimal value of β_U, β_τ and β_d through a cross validation process over training dataset. Moreover, we practically cannot search over all possible anonymization functions, hence we limit the possible functions to be a deep neural network and look for the optimal parameter set through training on the available datasets.

In the specific implementation of this paper, we choose mean squared error, *MSE*, as the distance function. *MSE* measures the distance between each raw data point and the corresponding transformed data point, hence it limits the anonymizer making apply hug distortion to data points. To approximate the required mutual information terms and based on the adversarial training approaches, we reformulate the optimization problem in (1) as a neural network optimization problem that is designed for training a convolutional auto-encoder (CAE).

Definition 2. *Let $A(\mathbf{X}|\theta)$ be a neural network, with parameter set θ, that gets the input vector $\mathbf{X}$ and transforms it to a same dimensional output vector $\mathbf{X}'$. The **optimizer** searches in the space of all the* *possible parameter sets, Θ, to find optimal parameter values:*

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \, \beta_U I(\mathbf{U}; A(\mathbf{X}|\theta)) - \beta_T I\big(\mathbf{T}; A(\mathbf{X}|\theta)\big) + \beta_d MSE(\mathbf{X}, A(\mathbf{X}|\theta)) \tag{2}$$

where, $\mathcal{A}(\cdot;\theta^*)$ is the optimal estimator for general $\mathcal{A}(\cdot)$ in (1).

To look for the optimal parameter set θ^* we use backpropagation using stochastic gradient descent. For this purpose we define a multi-objective loss function in section 3.3.

Figure 2 shows the high-level architecture for training a CAE to be used as the sensor data anonymizer. The Encoder maps $\mathbf{X}$ into an identity-agnostic latent representation $\mathbf{Y}$ by getting feedback from a pre-trained classifier, the Encoder Regularizer, which penalizes the Encoder if it captures information corresponding to the $\mathbf{U}$ into $\mathbf{Y}$. The Decoder aims to output a same-dimensional reconstruction of the input, $\mathbf{X'}$, from the $\mathbf{Y}$, and gets feedback from other pre-trained classifiers which we call them Decoder Regularizer, and Activity Regularizer, respectively.

Figure 3 shows the overall architecture, and Figures 4, 5 show the details of each neural network model that we have implemented in our experiments. *EncReg* (Encoder Regularizer) and *ActReg* (Activity Regularizer) share the same architecture as *DecReg* (Decoder Regularizer). The only differences are that the shape of input for *EncReg* is 32, instead of 128, and the shape of softmax output for *ActReg* is 4, instead of 24.

3.2 The training

Figure 6 summarizes the training of the CAE for an activity recognition task. We have two privacy regularizers, *EncReg* and *DecReg*, the task regularizer, *ActReg* for τ_a, and the distortion regularizer, a loss function that constrains the allowed distortion on the data.

Since convolutional layers capture locally autocorrelated and translation invariant patterns in time series data very well [15], *EncReg*, *DecReg*, and *ActReg* are convolutional neural network classifiers trained by categorical cross-entropy loss function [36] (see Figure 5). *EncReg* learns to identify users by getting the latent representation produced by the CAE (the input is $\mathbf{Y}$ and the output is the identity label, $\mathbf{U}$). *DecReg* learns to identify users by getting the reconstructed data produced by the CAE, $\mathbf{X'}$, as input (here the output is the identity labels, $\mathbf{U}$, too). *ActReg* learns to recognize the activity and gets the reconstructed data, $\mathbf{X'}$, as input and the activity label, $\mathbf{A}$, as output. Finally, the distortion regularizer, gets the original data, $\mathbf{X}$, and the reconstructed one, $\mathbf{X'}$, to calculate pointwise *mean squared error* as the distance function to measure the amount of distortion[5].

Instead of just training on a single epoch, like what is usual in adversarial training [11], all the classifiers here should be trained for some epochs, e, on the entire dataset to converge to suboptimal information estimators for use in the next step. The reason is, the game here is not to learn the exact data distribution, but to transform data from an identity-centric sample space which is so informative about users' identity, to another activity-centric sample space which is just informative about the underlying activity. Therefore, each regularizer should at least converge to a suboptimal approximator of mutual information.

[5]One can choose any other distance metric based on the tasks.

After each iteration, we should evaluate the condition of the current CAE to see if it needs one more iteration or not. Generally speaking, we need some practical and reliable evaluation process over validation data to check the provided utility-privacy trade-off. We discuss more about some of these possible evaluation methods in section 4.4.

3.3 Multi-objective loss function

After each training round, we freeze the parameters of the regularizers during the training of the CAE (see Fig.2). The most important contributor to the training of the CAE is our proposed multi-objective loss function, L, which we define as:

$$L = \beta_U L_U - \beta_T L_T + \beta_d L_d, \tag{3}$$

where the regularization weights β_U, β_T, and β_d determine the utility-privacy trade-off [6]. L_T and L_d are *utility losses* that can be customized based on the app requirements, whereas L_U is an *identity loss* that helps the CAE remove user-specific signals.

L_T, the categorical cross-entropy loss function for classification[7], aims to preserve activity-specific patterns:

$$L_T = \mathbf{T} \log(\hat{\mathbf{T}}), \tag{4}$$

where $\mathbf{T} \in \{0,1\}^B$ is a one-hot B-dimensional vector that represents true activity label for $\mathbf{X}$; and $\hat{\mathbf{T}} \in [0,1]^B$, the output of a softmax function[8], is a B-dimensional probability distribution vector that predicts the activity labels.

L_d aims to keep $\mathbf{X'}_{\mathrm{ij}}$, the output of the CAE, as similar as possible to the input $\mathbf{X}_{\mathrm{ij}}$:

$$L_d = \sum_{i=1}^{M} \sum_{j=1}^{W} (\mathbf{X}_{\mathrm{ij}} - \mathbf{X'}_{\mathrm{ij}})^2. \tag{5}$$

L_U is the most important term of our multi-objective loss function:

$$L_U = -\Big(\mathbf{U} \log(1 - \hat{\mathbf{U}}) + \log\big(1 - max(\hat{\mathbf{U}})\big)\Big), \tag{6}$$

where $\mathbf{U}$ is a one-hot N-dimensional vector that represents the true identity label for $\mathbf{X}$ and $\hat{\mathbf{U}} \in [0,1]^N$ is the output of the softmax function, the N-dimensional probability distribution vector learned by the classifier (i.e. the probability of each user label, given the input).

A trivial anonymization function would transform data of user 1 consistently into the data of user 2 (and vice versa). However, this function would satisfy only the first element of L_U. A desirable anonymization cannot allow a mapping between any users to any other users and we want our anonymization mechanism to transform $\mathbf{X}$ into $\mathbf{X'}$ in such a way that no adversarial model can confidently predict $\mathbf{U}$ from $\mathbf{X'}$. We achieve this objective by maximizing a measure of the difference between two distributions, i.e. the cross entropy, between the prediction, $\hat{\mathbf{U}}$, and the true identity, $\mathbf{U}$. Eq. 6 minimizes the cross-entropy between the true identity label and the regularizer's prediction of this label, as well as the

[6]In practice, we set it through a cross validation process and the sum of β values is equal to 1.

[7]For example for a fall detection app that monitors the stability of elderly people, a binary cross-entropy can be used instead of a categorical one.

[8]All entries of $\hat{\mathbf{T}}$ add up to 1.

```
1: procedure TrainCAE( $\mathbf{X}, \mathbf{U}, \mathbf{T}, e$)    ▷ $\mathbf{X}$: $M \times W$ sections from raw data, $\mathbf{U}$: identity label, $\mathbf{T}$: activity labels, and $e$ number of epochs.
2:     CAE (Encoder+Decoder) ⟵ Random Initializing;
3:     CAE ⟵ Train on $\mathbf{X}$ as both input and output for $e$ epochs;
4:     $\mathbf{Y}$ ⟵ Encoder($\mathbf{X}$);    ▷ $\mathbf{Y}$ is the extracted latent representation from the raw data.
5:     $\mathbf{X}'$ ⟵ $\mathbf{X}$;    ▷ Keep raw data intact to use it for evaluation in each iteration.
6:     EncReg, DecReg, ActReg, CAE ⟵ Random Initializing;
7:     do
8:         EncReg ⟵ Train on $\mathbf{Y}$ as input and $\mathbf{U}$ as output using categorical cross-entropy as loss function, for $e$ epochs;
9:         EncReg ⟵ Train on $\mathbf{X}'$ as input and $\mathbf{U}$ as output using categorical cross-entropy as loss function, for $e$ epochs;
10:        ActReg ⟵ Train on $\mathbf{X}'$ as input and $\mathbf{T}$ as output using categorical cross-entropy as loss function, for $e$ epochs;
11:        Freeze parameters of EncReg, DecReg, and ActReg;
12:        CombinedModel ⟵ The combination of CAE(Encoder+Decoder), EncReg, DecReg, and ActReg based on Figure 3;
13:        CombinedModel ⟵ Train on $\mathbf{X}'$ as input and $\mathbf{U}$, $\mathbf{U}$, $\mathbf{T}$, and $\mathbf{X}$ as outputs using $L_U$, $L_U$, $L_T$, $L_d$ as loss
                   functions, repectively, for $e$ epochs (Figure 2);
14:        $\mathbf{Y}$ ⟵ Encoder($\mathbf{X}$);
15:        $\mathbf{X}'$ ⟵ Decoder($\mathbf{Y}$);
16:        Unfreeze parameters of EncReg, DecReg, and ActReg;
17:    while it satisfies the convergence conditions;
18:    return CAE;    ▷ The final CAE that can be used as anonymizer on users' devices.
```

Figure 6: The adversarial regularization procedure to train a sensor Data Anonymizer; $\mathcal{A}(\cdot, \theta^*)$ in (2)

maximum value of the predicted identity vector, $\hat{\mathbf{U}}$. The derivation of the third term of the multi-objective loss function, L_U, is presented in the next section.

3.4 Derivation of the identity loss

In Eq. (2), $\mathbf{X}'$ is the transformed version of a user raw data whereas $\mathbf{U}$ is a latent random variable, drawn from $\mathbf{X}$, that determines the identity of the user who generates $\mathbf{X}$. Our goal is to make $\mathbf{U}$ and $\mathbf{X}'$ independent of each other.

Given $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{X}'$ are conditionally independent. Therefore the transformation $\mathbf{U} \to \mathbf{X} \to \mathbf{X}'$ is Markovian [26] and, based on the data processing inequality [3], the following holds:

$$I(\mathbf{U}; \mathbf{X}) \geq I(\mathbf{U}; \mathbf{X}'). \tag{7}$$

Processing data with a function f that aims to infer the identity of a user does not increase the available information. Therefore

$$I(\mathbf{U}; \mathbf{X}') \geq I(\mathbf{U}; f(\mathbf{X}')), \tag{8}$$

thus ensuring that when we reduce the mutual information between the user's identity and their released data, the mutual information between their identity, defined as a random variable (see Eq. (2)), and the outcome of any other adversarial function over the released data will also be reduced, at least by the same amount.

Using entropy, $H(\cdot)$, the mutual information can be defined as

$$I(\mathbf{U}; \mathbf{X}') = H(\mathbf{U}) - H(\mathbf{U}|\mathbf{X}'). \tag{9}$$

As the entropy is always non-negative and since we cannot control $H(\mathbf{U})$, to minimize $I(\mathbf{U}; \mathbf{X}')$, we can maximize $H(\mathbf{U}|\mathbf{X}')$, the conditional entropy between identity variable and the transformed data, which can be expressed as:

$$H(\mathbf{U}|\mathbf{X}') = H(\mathbf{U}, \mathbf{X}') - H(\mathbf{X}'). \tag{10}$$

To reduce the entropy of $\mathbf{X}'$ independently of any other latent variables, one could simply coarse-grain (e.g. downsample) the data. However, blindly minimizing $H(\mathbf{X}')$ can also lead to a substantial utility loss. Therefore, we focus on maximizing $H(\mathbf{U}, \mathbf{X}')$.

Let $p(\mathbf{U}, \mathbf{X}')$ denote the joint distribution of $\mathbf{U}$ and $\mathbf{X}'$; and $S_{\mathbf{u}}$ and $S_{\mathbf{X}'}$ be the supports of $\mathbf{U}$ and $\mathbf{X}'$, respectively. Then

$$H(\mathbf{U}, \mathbf{X}') = -\int_{S_{\mathbf{u}}} \int_{S_{\mathbf{X}'}} p(\mathbf{U}, \mathbf{X}') \log p(\mathbf{U}, \mathbf{X}'). \tag{11}$$

Since we cannot calculate the joint entropy directly, we need an estimator for $H(\mathbf{U}, \mathbf{X}')$. When labeled data is available $\mathbf{X}'$ can be used as input to predict $\hat{\mathbf{U}}$ as an estimation of $\mathbf{U}$. We can therefore reformulate the problem of maximizing the joint entropy, $H(\mathbf{U}, \mathbf{X}')$ into one of maximizing the cross entropy between the true variable $\mathbf{U}$ and the predicted variable $\hat{\mathbf{U}}$:

$$H_{\hat{\mathbf{U}}}(\mathbf{U}) = -\int_{S_{\mathbf{X}'}} \mathbf{U} \log \hat{\mathbf{U}}. \tag{12}$$

Here, we consider a multiclass classification problem, where $\mathbf{U}$ is a one-hot N-dimensional vector that represents true label, and $\hat{\mathbf{U}}$ is an N-dimensional probability distribution vector which is learned by the classifier. Therefore, the empirical cross entropy formula for a sample data $\mathbf{X}'$ of user k is:

$$-\mathbf{U} \log \hat{\mathbf{U}} = -\log \hat{\mathbf{U}}[\mathrm{K}] \tag{13}$$

where $\hat{\mathbf{U}}[\mathrm{k}]$ is the k-th element of the vector predicted by the classifier. Finally, since $\hat{\mathbf{U}}[\mathrm{k}] \in [0, 1]$, maximizing $-\log \hat{\mathbf{U}}[\mathrm{k}]$ is equivalent to minimizing $-\log(1 - \hat{\mathbf{U}}[\mathrm{k}])$.

Therefore, generalizing, the first term of Eq. (6), $\mathbf{U} \log(1 - \hat{\mathbf{U}})$, estimates the mutual information, $I(\mathbf{U}; \mathbf{X}')$. In fact, by forcing the CAE to minimize this value, we minimize the amount of user-identifiable information included in the $\mathbf{X}'$.

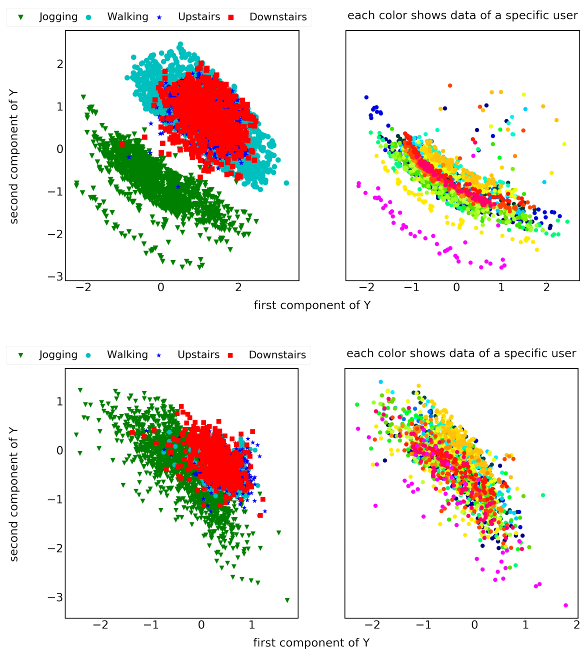

Figure 7: Latent representation, Y, in 2D of the 64D data of gyroscope. (Left column): samples of four different activities. (Right column): Jogging data for different users. (Top row): raw data. (Bottom row): data transformed by the CAE.

3.5 Examples

Figure 7 shows the result of using a convolutional auto-encoder as unsupervised feature extractor to represent 64D vectors of the gyroscope data, **X**, into 2D vectors, **Y**, by considering *MSE* as a measure of the reconstruction error. In the top plot of Figure 7 we see the latent representation of raw gyroscope data extracted by the bottleneck of the model. The distribution of **Y**, has useful information to distinguish not only the activities, but more also the user. Only using dimensionality reduction methods cannot ensure anonymization even if we extremely reduce the dimensions. Figure 7, bottom shows the latent representation of the anonymized data by our method. After transformation, the data for different users are more outspread and users are less separable from each other, but it preserves the jogging activity from the remaining well separable, like in the original data space[9].

Figure 8 shows the raw data versus the transformed version of a sample time window. The CAE obscures most of the patterns but still keeps some differences among data of different activities. Figure 9 compares the spectrogram of raw and transformed data for a user in the test dataset. One can notice that the CAE introduces new periodic component in the data and obscure some of the original ones, and they differ across the activities.

[9]This is an extreme representation of the data. In higher dimensions classifiers are able to find more discerning patterns in the data to distinguish other activities.

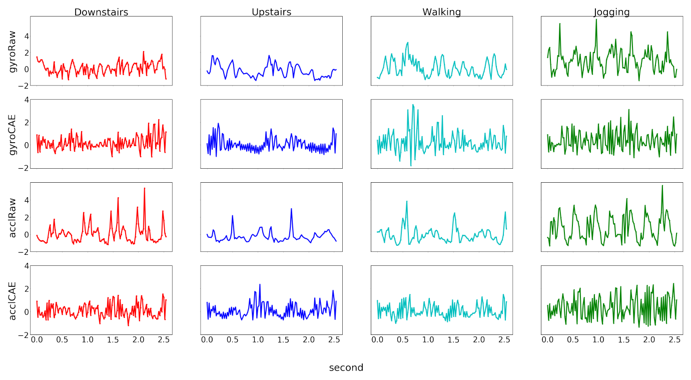

Figure 8: Raw data versus the output of CAE for gyroscope and accelerometer

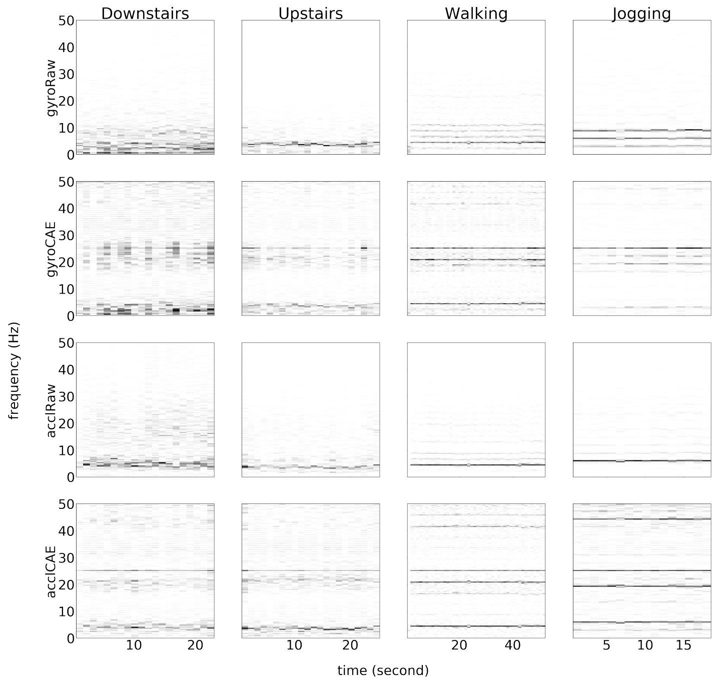

Figure 9: Spectrogram of raw data versus the output of CAE for one user in the test dataset.

4 EVALUATION

We assume an adversary has access to the training dataset and knows the anonymization mechanism. Furthermore, the adversary knows the users' physical and demographic attributes, so they can take advantage and use data of the most similar user in the public dataset for their adversarial purposes.

To evaluate the effectiveness of the CAE as a data anonymizer, we compare the trade-off between recognizing user's activity versus their identity, and compare with two baseline methods and the method that have been proposed in [9] which we call it REP. We

num. of users	24 (14 males -10 females)
sample rate	50 Hz
sensors	gyroscope accelerometer
Features	rotationRate (x,y,z) userAcceleration (x,y,z) gravity (x,y,z) attitude(roll, pitch, yaw)
Activities (num of trails)	downstairs (3 trials) upstairs (3 trials) walking (3 trials) jogging (2 trials) sat (2 trials) stand-up (2 trials)

Table 1: Details of the MotionSense datasets[19].

measure the extent to which activity recognition suffers from the anonymization compared to having access to the raw data[10].

4.1 Experimental Setup

The only requirement of our proposed method is to prepare a public dataset of motion sensor time series for training purpose. The dataset includes time series data generated by accelerometer and gyroscope sensors collected with an iPhone 6s kept in the participant's front pocket. A total of 24 participants in a range of gender, age, weight, and height performed 6 activities in 15 trials in the same environment and conditions (see Table 1).

We consider two methods of dividing the dataset into training and test sets, namely *Subject* and *Trial*. For Subject, we put all data of 4 of the users in the dataset, 2 females and 2 males, as test data and the remaining 20 users' data as training. Hence, after training the CAE, we evaluate the model on a dataset of new unseen users. For Trial, we put one trial data of each user as test data, and the remaining trials of that user's data as training. For example, we have three different walking trials for every user, we consider one trial as test and the other two as training; same for other activities. In both cases, we put 20% of training data for validation during the training phase. We repeat each experiment 5 times and report the mean, and the standard deviation. For all the experiments we use the magnitude value for both gyroscope and accelerometer.

To prepare the dataset ready for training, we have to choose the length of the time window, W and a step size for this rolling-window to move ahead over the time series. To be consistent in the process of encoding data into a lower-dimensional representation and then decoding it to the original dimension, it is better to set

[10]All the code and data used in this paper is publicly available and can be obtained from: https://github.com/mmalekzadeh/motion-sense

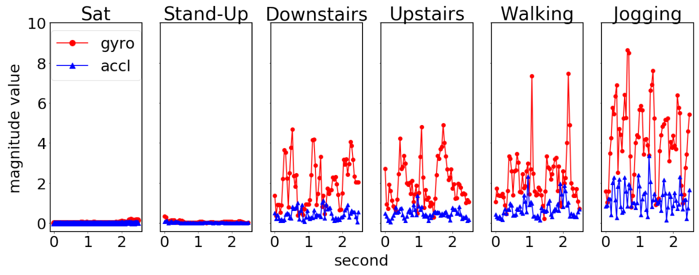

Figure 10: Magnitude value of data 2.5 seconds sample points of accelerometer (accl) and gyroscope (gyro), for six different activities for a specific user.

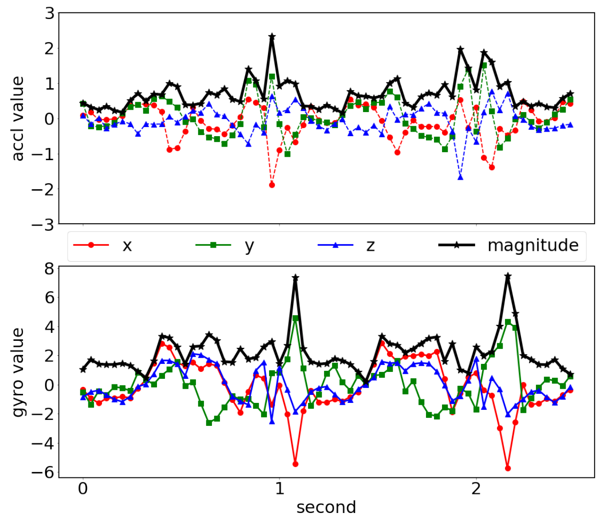

Figure 11: Value of each axis versus magnitude value for accelerometer (top) and gyroscope (bottom). Sample points from 2.5 seconds of walking for a specific user (walking section of Fig 10).

W a value that is a power of 2. As we discussed in section 4.2, the bigger the W, the lower the possibility of taking advantage of the correlation among the successive windows by adversaries. But larger window sizes lead to more delay for real-time applications and less data utility. Here, for all the experiments, we set $W = 128$ which equals to 2.56 seconds. We also set $S = 10$ as the step size of the rolling-window.

For all the regularizers, *EncReg*, *DecReg*, and *ActReg*, we use 2D convolutional neural networks. To prevent overfitting to training data, we put a Dropout [30] layer after each convolution layer. We also use L2 regularization to penalize weights with large magnitudes, and consequently this restriction forces classifier to learn features that are more relevant to the prediction.

4.2 Sensor Data Characteristics

Here we discuss some characteristics of motion sensor data that are important to be considered in design and evaluate of sensor data anonymizer mechanisms.

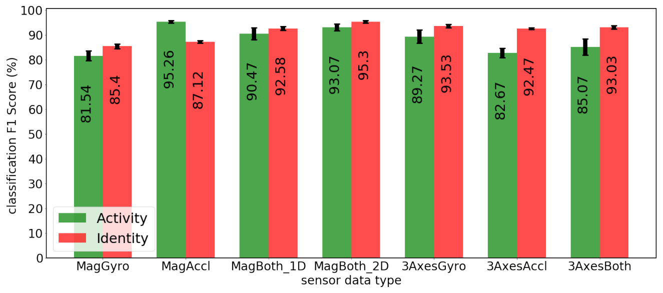

Figure 12: The effect of sharing different sensor data types in recognizing activity and identity on the corresponding test dataset. Mag **means magnitude,** Both **means both gyro and accl,** 1D **and** 2D **are the dimension of convolution filter, respectively.**

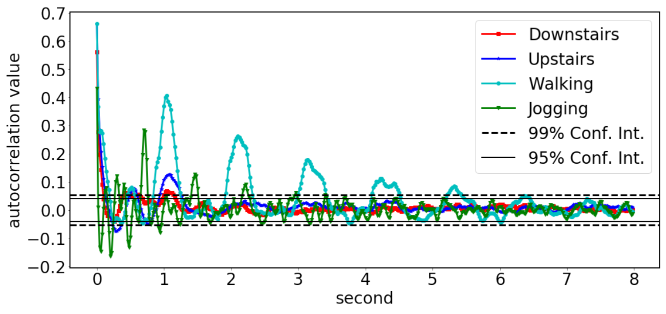

Figure 13: Autocorrelation plot of the accelerometer data for different activities averaged over all the users data. Correlation values outside the confidence interval lines are statistically significant, that means the results are reliable and not attributed to chance.

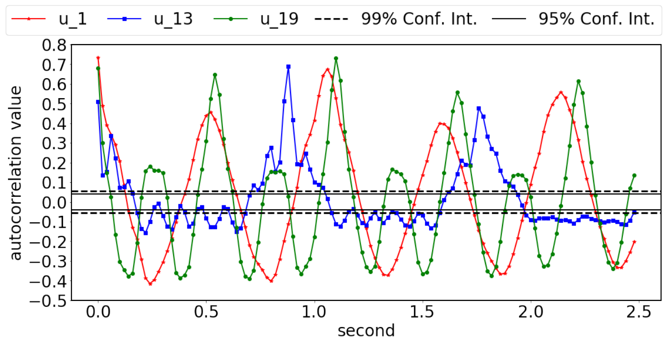

Figure 14: Autocorrelation of the accelerometer data for the walking activity for three different users.

Differences. Time series of gyroscope and accelerometer sensors give us insights into user's current state. Figure 10 compares the magnitude value of two sensors when the user is in six different states. Discerning patterns among activities can be seen in the data, however distinguishing motionless activities from each other, Sat and Stand-Up, is quite hard. We observed that the only data that is informative to distinguish motionless activities from each other is the value of gravity axes which determines the phone is being held in a vertical or horizontal direction. Users can only be distinguished from each other by observing their motion activities. Therefore, we do not consider motionless activities in our experiments.

Dimensions. Motion sensors measure the instant rotation and acceleration of the device, in all the three dimensions. As shown in Figure 11, there is a very informative correlation between the magnitude of these two kind of independently collected measurements. We see that both sensors almost follow each other, especially for the peaks and periodicity of the magnitude value, whereas a correlation among axes is less obvious.

Combinations. In Figure 12 we run an experiment to see: First, the effect of only sharing the magnitude value of each sensor with the app than sharing the exact value of each axis (x, y, z). Second, the effect of sharing the data of only one of these sensors than sharing both. It is very important to know how much information about user's identity can be extracted form the correlation between accelerometer and gyroscope. In Figure 12 we compare seven different settings of data sharing by calculating the F1 score for a deep convolutional neural network as the classifier[11]. For activity recognition, we consider Subject setting, but for identity recognition we use Trial setting.

One observation is, we can achieve equal and even better accuracy for activity recognition by only feeding the magnitude value to the model than all three axes, but for identity recognition we can take advantage of having exact values of each axis. Another observation is, when we use a 2D convolutional filters, which means the classifier model consider correlation among the inputs, than 1D filters that process each input time series separately, a better result is achieved for both activity and identity. Hence, a good anonymization mechanism should consider both inter-correlations, among adjacent values generated by one sensor, and intra-correlations, among coincident values generated bt both sensors. Based on this analysis, we use the MagBoth_2D case (sharing magnitude value of both gyro and accl) for all the experiments in the rest of the paper.

Correlations. Figure 13 shows the autocorrelation function (ACF)[12] for the magnitude of accelerometer data for different activities. We take the autocorrelation of 45 seconds of data for all users and average over these to get the ACF for all users. We see that motion data is a periodic time series, but each activity has a different period. Walking shows the highest correlation and after that we have jogging, upstairs, and downstairs, respectively. One interesting point is about the distance between two peaks which can be related to the stride intervals[13]. Another observation is that there are strong correlations among the points inside a time window of two seconds of the data. As a result, a good anonymization mechanism should consider a time window of at least 2 seconds to be sure that it takes these correlations into account. However, the correlations go under the confidence interval after about 5 seconds and this is very helpful for real-time situations where we cannot consider a big time window.

[11]By the similar architecture described in Figure 5

[12]It plots the autocorrelation for data points at varying time lags.

[13]Note that the sample rate of the dataset is 50Hz.

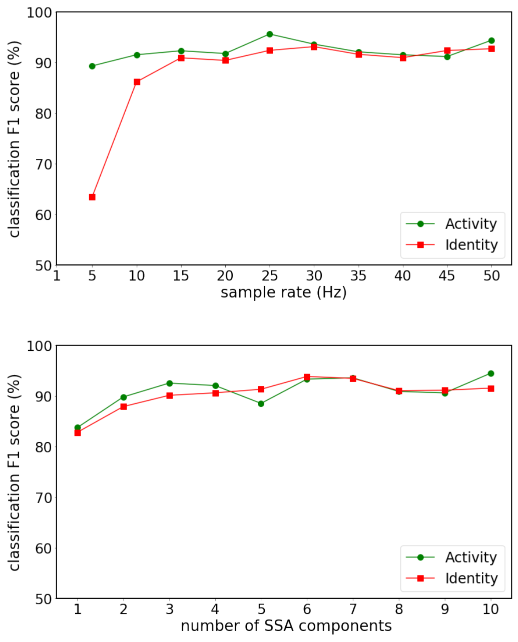

Figure 15: The classification accuracy of a deep convolutional neural network for both activity and identity recognition on test data. (Top) when data is resampled to another rate. (Bottom) when data is reconstructed by a subset of components (from a total of 50 components), ordered from largest to smallest by corresponding singular values.

What is more important for us is what we see in Figure 14 that shows the autocorrelations of same activity for three different users. The differences should be directly related to their characteristics. For example, the heavier the user [14], the bigger intervals between two peaks. This user-identifiable pattern seems a challenging feature of the sensory data to obscure before sharing. We can intuitively say why baseline methods like downsampling the data cannot effectively hide the user's identity, because these periodic patterns can be approximately retrieved when we apply the upsampling by a some effective filters.

4.3 Baseline Methods

Here we explain two baseline approach to coarse-grain time series data: *resampling* and *singular spectrum analysis*. In the next section, we compare the outputs of our proposed method with the outputs of these methods.

Resampling. In time series analysis literature there are several ways of resampling a series from its original sample rate to a target sample rate [25]. Time series of mobile sensor data for activity recognition all contain periodic components. Resampling by Fast Fourier Transform (FFT) provides better results for periodic time series. Therefore, we choose it as the resampling algorithm [15]. Here the naive idea is reducing the richness of the data to the extent that it contains useful information for recognizing the underlying task while it no longer contains identity-centric patterns (e.g. high-frequency components).

The top plot in Figure 15 presents the classification accuracy over test dataset when we feed the classifier a downsampled version of the original data. In this experiment we train a fix model for all the sample rate values to have fair comparisons. Result shows the impact of coarse-graining on activity recognition task can be ignored for a sample rate bigger than 20Hz. But, surprisingly, even by having a 5 samples per second (5Hz) dataset, we can distinguish 24 different users from each other by more than 60% accuracy.

Singular Spectrum Analysis. SSA [5] is a method for decomposing time series data into a collection of interpretable components such as trend, periodic, and structureless (or noise) components. SSA has a wide range of applications, because it is a model-free technique and does not assume stationarity-type conditions. It is based on the singular value decomposition (SVD) of the trajectory matrix constructed upon the time series and only has a window length parameter that specifies the number of components. In our case, we can decompose each $\mathbf{X}$, into a set of D components, $\{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_d\}$, such that the original time series can be exactly recovered by summing together all of the extracted components:

$$\mathbf{X} = \sum_{i=1}^{D} \mathcal{X}_i \tag{14}$$

One interesting aspect of SSA is the $\mathcal{X}_i$'s are arranged in descending order according to their corresponding singular value. Thus, we explore the idea of *incremental reconstruction* as a coarse-graining method, in the bottom plot of Fig.15. It shows training a classifier on the reconstruction with only the first components (of a total 10 extracted components) can achieve a more than 80% accuracy for both activity and identity recognition. This results give us a hint that each SSA component contain information about both activity and identity.

4.4 Discussion

We compare the output of the CAE with baseline methods and with *Rep* [9], which only consider the latent representation and locates the regularizers to only monitor $\mathbf{Y}$, and does not take $\mathbf{X}'$ into account (Figure 2).

Classification Accuracy. We train the classifier on both the original and the anonymized training dataset, and then use it for inference on the test data. Here we use the Subject setting, thus the test data includes data of new unseen users. Table 2 shows that the average accuracy for activity recognition, for both Raw and CAE data is around 92%. Comparing to other methods, that decrease the utility of the data, we can preserve the utility and even make it a bit better in average. One reason is, the CAE shapes data such that an activity recognition classifier can learn better from the transformed data than the raw data. We have seen in section 3.5 that CAE introduces new frequencies into the transformed data and they are different for each activity.

To evaluate the degree of anonymity, we train a classifier in Trial setting over raw data and then feed it different types of transformed data. Table 2 shows that downsampling data from 50Hz to 5Hz reveals more information than using the CAE output in the original

[14]Here the user u_1 is the heaviest among the three.
[15]Specifically, we use "signal.resample" function of "SciPy" package.

info.	result type	raw (50Hz)	resample (10Hz)	resample (5Hz)	SSA (1,2)	SSA (1)	REP [9] (50Hz)	**CAE** (50Hz)
(I) act	mean F1	92.51	91.11	88.02	88.59	87.41	91.47	**92.91**
(subject)	var F1	2.06	0.63	1.85	0.91	0.89	00.87	**0.37**
(I) id	mean ACC	96.20	31.08	13.53	34.13	16.07	15.92	**6.98**
(trial)	mean F1	95.90	25.57	8.86	28.59	12.58	11.25	**1.76**
(II) id	mean Rank	0	7	9	7	9	6	**7**
(DTW)	var Rank	0	6	6	6	5	6	**5**

Table 2: The trade-off between utility for activity recognition (act) and privacy of identity (id). ACC and F1 means accuracy and F1 score of classification, respectively. The forth column shows the K-NN rank between 24 users.

frequency. This results show the CAE can effectively obscure user-identifiable information, such that even a model that have had access to users' original data cannot distinguish them after applying the transformation.

The k-Nearest Neighbors. To evaluate the efficiency of the anonymization with another unsupervised mechanism, we implement the k-Nearest Neighbors (k-NN) with Dynamic Time Warping (DTW) [27], which is one of the best methods for clustering time series data[16]. Using DTW, we measure the similarity between the transformed data of a target user k and the raw data of each user l, $\mathbf{X}^{\mathsf{l}}$, for all $\mathsf{l} \in \{1, 2, \ldots, \mathsf{k}, \ldots, \mathsf{N}\}$, including k itself. Then we use this similarity measure to find the nearest neighbors of user l and check the rank of k among them. Table 2 shows it is very difficult to find similarities between users' transformed data and their raw data as the performance of the CAE is almost the same as the basic methods and we have a constraint in Eq. (2) to keep the data as similar as possible to the original data.

5 CONCLUSION

We proposed a new multi-objective loss function to train convolutional auto-encoders (CAE) as sensor data anonymizers that run on personal devices. The proposed solution is important to ensure anonymization for participatory sensing [6], when individuals contribute data recorded by their personal devices for health and well-being data analysis. We also not only consider the feature extractor part of the neural network model (encoder) to remove user-identifiable features included in available training data, but we also force the reconstructor part (decoder) to shape final output independent of every user in the training set, so the final trained model will be a generalize model which can be used by to a new unseen user. Finally, we share with apps a data of the same dimension as the original data, thus our method can easily mediate existing sensors and apps.

As future work, we aim to measure the cost of running such local transformations on user devices; to conduct experiments on other use cases (i.e. different tasks); and to derive statistical bounds for the amount of the achived privacy.

[16] k-NN with DTW outperforms other methods in time series classification, except when considerable computation and implementation cost is acceptable for very small improvements [2].

ACKNOWLEDGEMENT

This work was supported by the Life Sciences Initiative at Queen Mary University London and a Microsoft Azure for Research Award. Hamed Haddadi was partially supported by the EPSRC Databox grant (Ref: EP/N028260/1). We would also like to thank Deniz Gunduz and Emiliano De Cristofaro for their constructive feedback and insights on the research.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. 308–318.
[2] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. 2017. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 31, 3 (2017), 606–660.
[3] Normand J Beaudry and Renato Renner. 2011. An intuitive proof of the data processing inequality. *arXiv preprint arXiv:1107.0740* (2011).
[4] Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.
[5] David S Broomhead and Gregory P King. 1986. Extracting qualitative dynamics from experimental data. *Physica D: Nonlinear Phenomena* 20, 2-3 (1986), 217–236.
[6] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. 2006. Participatory sensing. In *Workshop on World-Sensor-Web (WSW'06): Mobile Device Centric Sensor Networks and Applications*. 117–134.
[7] John Duchi, Martin J Wainwright, and Michael I Jordan. 2013. Local privacy and minimax bounds: Sharp rates for probability estimation. In *Advances in Neural Information Processing Systems*. 1529–1537.
[8] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. 2010. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM, 715–724.
[9] Harrison Edwards and Amos Storkey. 2016. Censoring Representations with an Adversary. In *International Conference in Learning Representations (ICLR2016)*.
[10] Jonas Gehring, Yajie Miao, Florian Metze, and Alex Waibel. 2013. Extracting deep bottleneck features using stacked auto-encoders. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 3377–3381.
[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.
[12] Jihun Hamm. 2017. Minimax filter: learning to preserve privacy from inference attacks. *The Journal of Machine Learning Research* 18, 1 (2017), 4704–4734.
[13] Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. 2017. Context-aware generative adversarial privacy. *Entropy* 19, 12 (2017), 656.
[14] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[15] Yann LeCun and Yoshua Bengio. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361, 10 (1995), 1995.
[16] Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. 2017. DEEProtect: Enabling Inference-based Access Control on Mobile Sensing Applications. *arXiv preprint arXiv:1702.06159* (2017).
[17] Chris YT Ma and David KY Yau. 2015. On information-theoretic measures for quantifying privacy protection of time-series data. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*. ACM,

427–438.

[18] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* (2015).

[19] Mohammad Malekzadeh, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. 2018. Protecting sensory data against sensitive inferences. In *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*. ACM, 2.

[20] Mohammad Malekzadeh, Richard G Clegg, and Hamed Haddadi. 2018. Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis. In *Internet-of-Things Design and Implementation (IoTDI), 2018 IEEE/ACM Third International Conference on*. IEEE, 165–176.

[21] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. 2011. Stacked convolutional auto-encoders for hierarchical feature extraction. In *International Conference on Artificial Neural Networks*. Springer, 52–59.

[22] N. Neverova, C. Wolf, G. Lacey, L. Fridman, D. Chandra, B. Barbello, and G. Taylor. 2016. Learning Human Identity From Motion Patterns. *IEEE Access* 4 (2016), 1810–1820. https://doi.org/10.1109/ACCESS.2016.2557846

[23] Seyed Ali Osia, Ali Taheri, Ali Shahin Shamsabadi, Kleomenis Katevas, Hamed Haddadi, and Hamid R Rabiee. 2019. Deep Private-Feature Extraction. *IEEE Transactions on Knowledge and Data Engineering*.

[24] Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang. 2012. ACCessory: password inference using accelerometers on smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*. ACM, 9.

[25] Ernesto Ramos. 1988. *Resampling methods for time series*. Technical Report. HARVARD UNIV CAMBRIDGE MA DEPT OF STATISTICS.

[26] Borzoo Rassouli and Deniz Gündüz. 2018. Optimal Utility-Privacy Trade-off with the Total Variation Distance as the Privacy Measure. *arXiv preprint arXiv:1801.02505* (2018).

[27] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11, 5 (2007), 561–580.

[28] Lalitha Sankar, S Raj Rajagopalan, and H Vincent Poor. 2013. Utility-privacy tradeoffs in databases: An information-theoretic approach. *IEEE Transactions on Information Forensics and Security* 8, 6 (2013), 838–852.

[29] Ali Shahin Shamsabadi, Hamed Haddadi, and Andrea Cavallaro. 2018. Distributed One-class Learning. In *IEEE International Conference on Image Processing (icip 18)*. IEEE.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[31] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. 2017. Privacy Loss in Apple's Implementation of Differential Privacy on macOS 10.12. *arXiv preprint arXiv:1709.02753* (2017).

[32] Ardhendu Tripathy, Ye Wang, and Prakash Ishwar. 2017. Privacy-Preserving Adversarial Networks. *arXiv preprint arXiv:1712.07008* (2017).

[33] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, 1096–1103.

[34] Jun Wang, Shubo Liu, and Yongkai Li. 2015. A review of differential privacy in individual data release. *International Journal of Distributed Sensor Networks* 11, 10 (2015), 259682.

[35] Fengjun Xiao, Mingming Lu, Ying Zhao, Soumia Menasria, Dan Meng, Shangsheng Xie, Juncai Li, and Chengzhi Li. 2018. An information-aware visualization for privacy-preserving accelerometer data sharing. *Human-centric Computing and Information Sciences* 8, 1 (2018), 13.

[36] Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. 2015. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 3995–4001. http://dl.acm.org/citation.cfm?id=2832747.2832806