ObscureNet: Learning Attribute-invariant Latent Representation for Anonymizing Sensor Data

Omid Hajihassnai
hajihass@ualberta.ca
University of Alberta
Edmonton, Canada

Omid Ardakanian
oardakan@ualberta.ca
University of Alberta
Edmonton, Canada

Hamzeh Khazaei
hkh@eecs.yorku.ca
York University
Toronto, Canada

ABSTRACT

In this paper, we introduce ObscureNet, an encoder-decoder architecture that effectively conceals private attributes associated with time series data generated by sensors in IoT devices, while preserving the information content of the original time series. Drawing on conditional generative models and adversarial information factorization, ObscureNet learns latent representations that are invariant to the user-specified private attributes. This allows for modifying the private attributes or generating them randomly before using the decoder to synthesize a new version of data. We present three approaches to alter private attributes at anonymization time, and show that non-deterministic approaches can prevent an adversary from re-identifying private attributes. We compare ObscureNet with the autoencoder-based anonymization methods proposed in the literature and other generative models in terms of the accuracy of sensitive and desired inferences. Our experiments on two human activity recognition datasets show that compared to the original data, the sensitive inference accuracy is reduced by 80.38% on average, while the desired inference accuracy is only reduced by 6.82%. Moreover, ObscureNet reduces the sensitive inference accuracy by an additional 13.48% on average compared to the best baseline method. We report the computation overhead of running ObscureNet on a Raspberry Pi, and corroborate that it can be used for real-time anonymization of sensor data.

CCS CONCEPTS

• **Computer systems organization** → *Sensor networks*; • **Computing methodologies** → *Learning latent representations*; • **Security and privacy** → **Privacy protections**.

KEYWORDS

Conditional generative models, sensor data anonymization

ACM Reference Format:
Omid Hajihassnai, Omid Ardakanian, and Hamzeh Khazaei. 2021. ObscureNet: Learning Attribute-invariant Latent Representation for Anonymizing Sensor Data. In *International Conference on Internet-of-Things Design and Implementation (IoTDI '21), May 18–21, 2021, Charlottesvle, VA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3450268.3453534

IoTDI '21, May 18–21, 2021, Charlottesvle, VA, USA

ACM ISBN 978-1-4503-8354-7/21/05…$15.00
https://doi.org/10.1145/3450268.3453534

1 INTRODUCTION

Embedded and networked sensors are becoming increasingly ubiquitous in our lives. Smart home devices, such as smart thermostats, security systems, and virtual personal assistants, integrate various sensors that gather a substantial amount of data from our living and work space. Wearable devices and mobile phones are equipped with multiple sensors producing time series of heart rate, skin temperature, blood oxygen level, etc. The large volume and diversity of the sensor data that has become available is a mixed blessing. On one hand, valuable insights can be generated through multimodal sensor fusion, enabling sophisticated monitoring and control applications, such as Human Activity Recognition (HAR) [14] and thermal comfort inference [9]. On the other hand, the collected data may contain specific patterns that disclose *private attributes* associated with people present in the monitored environment, such as their age, gender, race, weight, height, personality traits, and moods. These private attributes can be easily inferred from data using machine learning techniques [3, 24, 36], leading to serious privacy concerns.

The privacy concerns can be addressed by concealing private attributes before sharing sensor data with third-party applications. But doing this without reducing the *utility* of data, which is defined as the accuracy of desired inferences, is proven to be a difficult task. This is because the intersection between features (or patterns in time series data) that correlate to public and private attributes is typically nonempty. Thus, attempts to prevent sensitive inferences through down-sampling these features, adding noise to them, and masking them in the dataset can result in a significant loss of accuracy for desired inferences, hence a lower utility. This is an important barrier to widespread adoption of these privacy-preserving techniques as people by and large prefer convenience over privacy. A more practical and promising solution should maintain the accuracy of desired inferences, offering a better trade-off between privacy and utility.

In recent years, deep generative models, and in particular, encoder-decoder architectures have been used to manipulate natural images, e.g., to change the facial expression of a portrait [18, 19]. Inspired by the success of autoencoders and generative adversarial networks in the image generation task, they have been applied to anonymize data generated by IoT and mobile devices [11, 24, 25]. These approaches transform sensor data, such that the leak of private information is minimized. For example, an autoencoder is used in [25] to replace sections of time series that reveal private attributes with same-length sections that are neutral. In follow-on work [24], an autoencoder is trained in an adversarial fashion using a loss function that ensures maximum data distortion and minimum mutual

information between private attributes and synthesized data. However, the sensor data anonymized via both techniques can still be used to re-identify the user and their private information should the attacker pass a dataset with known private attributes through these autoencoders [11]. We refer to this as the re-identification attack. To prevent user re-identification, sensor data with private attributes must be transformed in a non-deterministic fashion, a direction we pursue in this work.

We propose an encoder-decoder architecture called ObscureNet. Unlike conventional autoencoders, which are suitable for learning unsupervised latent representations, ObscureNet learns latent representations that are constrained to be invariant to one or several private attributes. The invariance is obtained by incorporating the classification error of a classifier that predicts the private attribute given the latent representation in the loss function of ObscureNet. Thanks to adversarial training, we can build a network that reconstructs data such that it resembles the original sensor data, but fools the attribute classifier. Such latent representations are best for data anonymization using ObscureNet as the decoder network is fed the latent representation along with the private attribute. Hence, at anonymization time, we can modify the private attribute in a deterministic or probabilistic manner to ensure that the reconstructed data cannot be used to perform unwanted inferences. We show that modifying the private attribute in a non-deterministic fashion prevents the adversary from re-identifying this information by training another model given the output of ObscureNet when it is fed input data with known labels. Our contribution is threefold:

- We assess the efficacy of ObscureNet in concealing private attributes associated with sensor data by running experiments on two HAR datasets, namely MotionSense [24] and MobiAct [4]. We show that anonymization performed by ObscureNet reduces the accuracy of a sensitive inference by 13.48% compared to the best baseline method while maintaining the utility of data. Furthermore, we explain how the loss function of ObscureNet can be augmented to protect multiple private attributes.
- By tuning parameters of ObscureNet, we show that it is possible to navigate the privacy-utility trade-off. This enables users to trade utility for privacy (or vice versa) depending on the context and data that is being collected.
- We compare three different ways that ObscureNet can modify private attributes. We find that using a *stochastic vector* instead of the one-hot encoding of the private attribute, brings the sensitive inference accuracy to the level of a random guess and diminishes the possibility of re-identifying private information to a great extent. As a result, a sensitive inference model will not be needed at anonymization time and ObscureNet can easily run on IoT or edge devices to anonymize sensor data in real time. To show this, we run ObscureNet on a Raspberry Pi 3 model B and time its execution.

While the application of conditional deep generative models with information factorization to image synthesis has been explored in the literature (see for example *Fader Networks* [19]), to the best of our knowledge, these conditional models have not been extended and adapted to address the anonymization of time series data. Despite having a similar architecture to Fader Networks [19], ObscureNet utilizes a VAE instead of an autoencoder. We examine 3 modifiers that can be used with ObscureNet to anonymize sensor data. These modifiers have no counterpart in Fader Networks.

Previous work on sensor data anonymization using autoencoders has several shortcomings which are discussed in the next section.

2 RELATED WORK

Privacy is a major issue in sensor-rich environments. While pervasive sensing and inconspicuous data collection bring about new services, they can lead to major privacy concerns. As shown in previous work, voice data collected by the built-in microphone of virtual personal assistants and mobile phones, often without user's knowledge and consent, can be analyzed to infer their emotional state and mental health condition [3, 12]. Similarly, cameras embedded in mobile devices can collect sensitive and personal data, especially in indoor environments [37]. According to [32], many mobile apps extract information from camera data that users do not expect. This can turn into visual privacy leaks if the app is malicious, i.e., makes an unsolicited or intrusive inference on this data, or when this data is sent to third-party servers for storage and processing. Examples of intrusive inferences are gender inference performed by a fitness tracking app [24] and building occupancy detection using smart meter data [5].

Many different solutions have been proposed to date to protect user privacy, ranging from physically disrupting the data collection process when there is a risk of privacy infringement, to adopting access-control mechanisms that contain the use of private data by malicious apps, to transforming sensor data to conceal private information before it is accessed by apps. To disrupt data collection, as in [37] where an LED is used to generate flickering patterns, the user must have control over the sensor or have the ability to meddle with the environment. However, interfering with the data collection process is not always possible and most related work assumes that sensor data is already collected.

The privacy-preserving techniques can be broadly categorized into hardware and operating system-level solutions, and application-level solutions. The former category includes fine-grained access-control mechanisms and isolated execution environments [20, 26, 29]. These solutions are usually tailored to specific use cases and are incapable of preventing the leak of private information while maintaining the utility of data. The latter category includes federated learning frameworks [28], differential privacy algorithms [7, 31], cryptographic solutions based on homomorphic encryption and compressive sensing [1, 34], and privacy-preserving techniques that transform data to a subspace where private attributes can be easily identified and altered [8, 11, 24]. Federated learning addresses the problem of training a model given data from many users without transferring them to a server. It is not useful when a public dataset is available for training the inference model, which is the assumption we make in this work. Differential privacy algorithms endow users with plausible deniability, but are not suitable for anonymizing streaming sensor data on IoT devices. Existing methods based on homomorphic encryption are computationally expensive. Hence, privacy-preserving techniques that employ data transformation are more favorable for efficient data anonymization on IoT devices.

Deep generative models, such as Generative Adversarial Networks (GANs) [10], Autoencoders, and Variational Autoencoders (VAE) [17], are extensively used to generate a realistic version of an image which has a few differences with the original version (e.g., an image that has a different background color) [18, 19]. Apart from image synthesis, generative models have been utilized to produce synthetic time series datasets [2, 21, 22, 33]. For instance, in [2] a Wasserstein GAN is used to generate balanced and realistic sensor data for HAR. In a recent study [22], the authors propose a framework based on GAN, called DoppelGANger, to generate network time series data with 43% higher fidelity than other baselines. Moreover, variants of autoencoders are commonly used to learn useful representations, especially when multiple sensing modalities are present. For example, autoencoders are used in [27] to learn a shared representation between multiple modalities.

Inspired by advances in deep generative models, recent work on data anonymization [24, 25] uses autoencoders to reconstruct the input data such that private attributes are no longer identifiable. This approach provides a reasonable trade-off between utility and privacy by minimizing the leak of private information while preserving the information content of the input data. However, the data anonymized by these networks is shown to be susceptible to the re-identification attack [11]. To address this shortcoming, a probabilistic transformation technique is proposed in [11] which manipulates private attributes such that the anonymized data is less vulnerable to the re-identification attack. This approach has several disadvantages. It requires a central entity to collect, update, and propagate the mean of latent representations over time. Furthermore, the anonymization does not preserve the information content of the original data, reducing its utility.

To provide a better trade-off between privacy and utility, the privacy adversarial network (PAN) is introduced in [23]. PAN learns an encoder that generates features from the raw sensor data by combining adversarial training with generative and discriminative training. This work is different from ours as it aims to generate task-specific features in a privacy-preserving manner, whereas our goal is to construct an anonymized version of the sensor data. In [13], segments of the time series that can be used for sensitive inferences are black-listed, while other segments that can be used to make desired inferences are white-listed. The authors propose the Generative Adversarial Privacy (GAP) framework to offer a trade-off between utility and privacy. Replacement Autoencoder [25] builds on this idea by adding grey-listed inferences, i.e., non-sensitive inferences, to the white-listed and black-listed inferences introduced in [13]. These techniques are more suitable for replacing activities that are privacy intrusive, for example smoking and drinking. According to our evaluation (Section 6), race, gender or other private attributes cannot be obscured using these techniques as white-listed segments contain information about these attributes.

In [35], anonymization is performed through learning perturbations in transforming raw sensor data. The goal of these transformations are to reduce the sensitive inference accuracy while maintaining the accuracy of desired inferences. The authors refer to private attributes as *style* and public attributes as *content*. A transformation is used to map the style to random noise. We compare the anonymization performance of ObscureNet with the performance of this method in Section 6 and show that our method is superior.

None of the above techniques introduce structure into the latent representation of autoencoders and utilize it to control data attributes in the synthesis process. We show through ablation studies that learning latent representations in a supervised manner opens the door to various anonymization techniques which differ in how they modify the private attribute. It reduces the leak of private data to a great extent and provides a better trade-off between utility and privacy.

3 ENCODER-DECODER ARCHITECTURE

Let $\mathcal{X}$ be the domain of fixed-length embeddings of time series data generated by one or several sensors, and $\mathcal{Y}$ and $\bar{\mathcal{Y}}$ be respectively domains of private and public attributes that can be associated with embeddings in $\mathcal{X}$. Our dataset, $\mathcal{D} = \{(x_1, y_1, \bar{y}_1), ..., (x_m, y_m, \bar{y}_m)\}$ consists of m data embeddings, each denoted by x_i, and their corresponding private and public attributes denoted by y_i and $\bar{y}_i$. We assume this dataset is publicly available, and can be used by anyone to train models for desired and sensitive inferences[1]. We consider categorical attributes such as mood, activity, and gender. Hence, the private attribute takes value from $\mathcal{A} = \{a_1, \cdots, a_K\}$ and the public attribute takes value from $\mathcal{B} = \{b_1, \cdots, b_{\bar{K}}\}$.

We present three autoencoder architectures below, and give a mathematical derivation of the loss function in each case. The main distinction between these architectures lies in their ability to impose some structure into the latent space. We discuss different approaches to incorporate structure into the latent representations learned by a VAE, and explain how this structure helps change private attributes associated with sensor data. The reason we focus on autoencoders rather than GANs is that they represent the data distribution more faithfully and provide a simple way to map data to its latent representation, which can be manipulated to anonymize data.

3.1 Variational Autoencoder

A VAE is an autoencoder comprised of a probabilistic encoder and a probabilistic decoder which are instantiated as two neural networks. The probabilistic encoder $q_\theta(z|x_i)$ maps sensor data x_i (or an embedding of it) to a distribution (e.g., a multivariate Gaussian) over low-dimensional continuous latent representations from which x_i could have been generated. The probabilistic decoder $p_\phi(x_i|z)$ produces a distribution over x_i given its latent representation z. This model can be used to generate a new version of the sensor data denoted by $\tilde{x}_i$. Note that θ and ϕ are network parameters that can be learned jointly.

Instead of maximizing the marginal likelihood which is typically intractable, the VAE is trained to maximize a lower bound on the marginal log-likelihood which is known as the evidence lower bound (ELBO) [17]. This lower bound can be written for an individual data point denoted by x_i as follows:

$$\text{ELBO}_i(\phi, \theta) = \mathbb{E}_{z\sim q_\theta(z|x_i)} \log p_\phi(x_i|z) - \text{D}_{\text{KL}}\big(q_\theta(z|x_i)||p(z)\big) \quad (1)$$

The Kullback–Leibler (KL) divergence term in ELBO acts as a regularizer for the approximate posterior.

In the training phase, we maximize the sum of $ELBO_i$ over all samples in $\mathcal{D}$. This ensures that the encoder maximally preserves

[1] Preventing the membership inference attack is outside the scope of this paper.

the information content of the input data and the decoder produces data as close as possible to its original input.

Concealing private attributes with VAE: Suppose a VAE is trained on $\mathcal{D}$. If we knew which latent variable corresponds to a given private attribute, we would be able to modify this attribute before the new version of data is generated by the decoder. Unfortunately this is not possible because latent representations are learned in an unsupervised fashion. Hence, to perform anonymization we have to modify all latent variables. This is the idea of the mean manipulation technique proposed in [11]. To perform anonymization the authors calculate the mean latent representation of all samples in $\mathcal{D}$ which have the same pair of public and private attributes. These mean latent representations are then used to manipulate the latent representation of an input data before it is passed to the decoder.

The manipulation process proposed in [11] is based on two arithmetic operations in the latent variable space of the VAE. Let us denote the mean latent representation of all data points in $\mathcal{D}$ which have private attribute $y = a_i$ and public attribute $\bar{y} = b_j$ by $z_{a_i}^{b_j}$. Now consider the latent representation z_i corresponding to x_i which has private attribute a_i and public attribute b_j. The private attribute can be changed to a_k $(k \neq i)$ in the data generation process by subtracting $z_{a_i}^{b_j}$ from z_i and adding back $z_{a_k}^{b_j}$:

$$z_i' = z_i - z_{a_i}^{b_j} + z_{a_k}^{b_j}$$

The new latent representation is then used by the probabilistic decoder to produce $\tilde{x}_i$. Note that the target private attribute for the new version of data, i.e., a_k, can be chosen in a deterministic or probabilistic fashion.

We call this technique anonymization with a *general VAE* and use it as a baseline in Section 6. As we discuss in that section, the mean manipulation technique can lower the accuracy of intrusive inferences, but at the cost of reducing the accuracy of desired inferences. Moreover, tracking changes in the mean latent representation of a specific class of data points and propagating these changes to mobile and IoT devices performing anonymization is difficult.

3.2 Conditional Variational Autoencoder

While variational autoencoders are suitable for learning unsupervised latent representations of data, the learned latent variables cannot be explained or mapped to salient attributes of input data. Learning useful latent variables that correlate to specific attributes in the dataset has received a lot of attention in recent years [16, 18]. Several efforts have been made to date to incorporate structure into latent representations in a supervised or semi-supervised fashion. Related work such as [16, 30] introduces structure in the latent space by conditioning latent variables on the attributes. This can be accomplished by directly incorporating these features into the latent representation as in the conditional VAE (CVAE). Specifically, a CVAE conditions the encoder, the decoder, or both on random variables representing data attributes. Thus, the probabilistic encoder and decoder can be written as $q_\theta(z|x_i, c)$ and $p_\phi(x_i|z, c)$, where the condition c can be a certain attribute of input data which we wish to encode. For example, it can be the private or public attribute(s) associated with an individual data point in a labelled dataset.

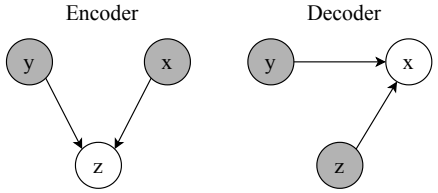

Figure 1: Graph diagram of encoder and decoder of a CVAE.

The variational lower bound of CVAE can be derived from (1). The lower bound for an individual data point can be written as:

$$\mathbb{E}_{z\sim q_\theta(z|x_i,c)}\log p_\phi(x_i|z,c) - \mathrm{D}_{\mathrm{KL}}\big(q_\theta(z|x_i,c)||p(z)\big) \qquad (2)$$

This objective is maximized using a stochastic optimization method to train the autoencoding model. In practice the condition and learned latent variables can be concatenated before they are passed to the decoder for reconstructing the data. Figure 1 shows the encoder and decoder of a CVAE.

Concealing private attributes with conditional models: We can use the CVAE framework as a basis for data anonymization. If the CVAE model is conditioned on the private attribute of data, we can manipulate the private attribute of the reconstructed data, $\tilde{x}_i$, through a simple modification of the condition variable. This allows us to change the condition of the CVAE from y_i to y_i' to change the private attribute of the reconstructed data from y_i to y_i'.

Consider a variant of this technique where a separate CVAE is trained for each public attribute. It helps address the data imbalance problem in training autoencoders as samples with different public attributes are not evenly distributed in $\mathcal{D}$. We call this technique anonymization with *attribute-specific CVAEs* and use it as a baseline in Section 6. Interestingly, the same technique can be used with a conditional autoencoder (CAE) where the loss function does not have the KL divergence term. We call this technique anonymization with *attribute-specific CAEs* and use it as a baseline in Section 6.

3.3 Adversarial Information Factorization

The condition variable in the CVAE model can be utilized to alter a certain attribute of the reconstructed data. However, the leak of information about this attribute to latent variables undermines the ability to change this attribute without affecting other aspects of the reconstructed data. This problem can be mitigated by enforcing and maximizing the disentanglement between latent variables and the condition variable. Specifically, learning attribute-invariant latent variables [19] can be achieved through the incorporation of an adversarial objective into the CVAE's ELBO expressed in (2). Similar to [6, 19], the condition variable in this case is not fed to the encoder to produce a latent representation; this changes the term $q_\theta(z|x_i, c)$ in (2) to $q_\theta(z|x_i)$.

Using adversarial training the networks can be trained so as to disentangle the condition from the latent representation [19]. Here, we train a neural network, Disc : $Z \rightarrow C$, that is trained in conjunction with the CVAE. The neural network Disc is trained to infer the true condition corresponding to each latent representation z. Moreover, the encoder is trained to undermine the accuracy of the adversarial model by learning latent variables that capture the

least possible amount of information about the condition. We use this idea in the design of ObscureNet which is described next.

4 OBSCURENET

ObscureNet is an encoder-decoder architecture that can be augmented with as many discriminator networks as there are private attributes. Each discriminator network predicts the probability distribution over one private attribute given the latent variables. Figure 2 shows the architecture of ObscureNet when there is only one private attribute associated with the sensor data that we wish to protect. ObscureNet conditions the decoder on the private attributes (similar to a CVAE) and performs adversarial information factorization to ensure that the learned latent representation is invariant to the private attributes.

In essence, by combining the best of conditional deep generative models and information factorization, ObscureNet can modify the private attributes without affecting the information about public attributes which are included in the synthesized time series data. An advantage of this design is that the obfuscation of private attributes reduces to a simple modification of the decoder's condition variables. This modification can be either deterministic or probabilistic as discussed in Section 4.2. We now describe how ObscureNet is trained given a dataset of samples with known private and public attributes. This training can be done in a cloud server, or on the IoT device if it has access to the public dataset used for training (i.e., $\mathcal{D}$). If the network is trained in the cloud, the weights and parameters of the encoder and decoder networks must be sent to IoT devices that run ObscureNet locally to anonymize their data.

4.1 Adversarial Training

We first describe how the ELBO of a CVAE is modified for training an encoder-decoder architecture that learns a latent representation containing little or no information about the private attributes. We then outline the process of training ObscureNet using an iterative minimax algorithm [19]. Without loss of generality and for ease of presentation, in the following, we consider the case that there is only one private attribute we want to conceal. Should there be more private attributes, the decoder must be conditioned on all these attributes and the adversarial loss function should include the cross-entropy loss of multiple discriminators.

Figure 2 shows the architecture of ObscureNet and the flow of gradients through the networks. In ObscureNet, the CVAE is augmented with a discriminator network which outputs $P_\eta(y|z)$, i.e., private attribute class-membership probabilities given the latent representation of input data. Here, η represents trainable parameters of the discriminator. If the private attribute and learned latent variables are completely disentangled, the discriminator would not be able to predict the private attribute.

The discriminator network can be trained using binary or categorical cross-entropy loss depending on whether the corresponding private attribute is binary (e.g., male or female) or categorical (e.g., weight). The loss of the discriminator network, $\mathcal{L}_{disc}$, is:

$$\mathcal{L}_{disc}(\eta|\theta) = -\frac{1}{m}\sum_{(x,y)\in\mathcal{D}} \log P_\eta(y|z) = -\frac{1}{m}\sum_{(x,y)\in\mathcal{D}} \log P_\eta(y|G_\theta(x)), \tag{3}$$

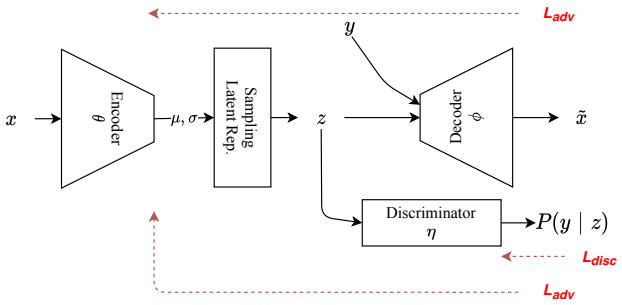

Figure 2: Training of ObscureNet

where m is the number of samples in $\mathcal{D}$ and $G_\theta(x)$ is a function that compactly represents both the encoding of input and sampling z from a multivariate Gaussian distribution. This is similar to the reparameterization trick used in [17].

The adversarial loss function can be written as:

$$\begin{aligned}\mathcal{L}_{adv}(\theta,\phi|\eta) = -\frac{1}{m}\sum_{(x,y)\in\mathcal{D}} \Big(&\mathbb{E}_{z\sim q_\theta(z|x)}\log p_\phi(x|z,y) \\ &- \beta\, D_{\mathrm{KL}}(q_\theta(z|x)||p(z)) \\ &- \alpha\, \log P_\eta(y|G_\theta(x))\Big)\end{aligned} \tag{4}$$

It combines the discriminator loss (3) with CVAE's ELBO from (2). We use the standard scalarization method and introduce weights that determine the relative importance of different terms in the adversarial loss function. The weights α and β are respectively assigned to the discriminator loss and the KL-divergence term. We treat these weights as hyperparameters and tune them in Section 6 to navigate the trade-off between utility and privacy.

ObscureNet is trained using an iterative algorithm, described in Algorithm 1. The discriminator is trained to predict y given z while the CVAE is trained to minimize the accuracy of the discriminator in addition to minimizing the loss function of the CVAE. While training the discriminator with parameter η, gradients are stopped from updating the CVAE network parameters: θ and ϕ. In the same respect, gradients are stopped from updating the parameter of the discriminator network when training the CVAE.

Algorithm 1 shows the minibatch gradient descent for training ObscureNet, where B is the size of the minibatch. Both forward and the backward passes of ObscureNet's adversarial training can be seen in the pseudocode.

We remark that ObscureNet utilizes a different set of encoder and decoder networks for each public attribute. Each pair of these networks are trained separately using only samples in $\mathcal{D}$ that have the same public attribute. We argue that this helps to reduce the number of layers and neurons in the neural networks, making it easier to run ObscureNet on resource-constrained devices[3].

4.2 Anonymization with ObscureNet

After training the networks in an adversarial setting, we use them to perform anonymization before sharing sensor data with third-party applications that run locally or uploading it to cloud servers

Algorithm 1: Training ObscureNet

Data: Training dataset $\mathcal{D}$, learning rate λ, minibatch size B
Result: Network parameters θ, ϕ, η
$\theta, \phi, \eta \leftarrow$ initial values
repeat
  Sample a minibatch: $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_B\}$
  `/* pass samples through the networks */`
  $\boldsymbol{\mu}, \boldsymbol{\sigma} \leftarrow \text{Enc}(\boldsymbol{x}; \theta)$
  $\boldsymbol{\epsilon} \sim N(0, I)$
  $\boldsymbol{z} \leftarrow \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
  $\tilde{\boldsymbol{x}} \leftarrow \text{Dec}(\boldsymbol{z}, y; \phi)$
  $P(y|\boldsymbol{z}) \leftarrow \text{Disc}(\boldsymbol{z}; \eta)$
  Estimate gradients of minibatch
  `/* Update parameters using gradients */`
  $\eta \leftarrow \eta$ - $\lambda \nabla_\eta \, \mathcal{L}_{disc}(\eta|\theta)$
  $\{\theta, \phi\} \leftarrow \{\theta, \phi\}$ - $\lambda \nabla_{\{\theta,\phi\}} \, \mathcal{L}_{adv}(\theta, \phi|\eta)$
until convergence of parameters (θ, ϕ, η)

which host these applications. In particular, sensor data embeddings are passed through ObscureNet which obscures their private attributes, i.e., generates a new version of each embedding with private attribute(s) that might be different from the original version. This process is depicted in Figure 3. Keep in mind that the encoder and decoder are the same networks trained using samples in $\mathcal{D}$. As it can be seen, in addition to encoder and decoder networks, we take advantage of a classification model that is trained separately to identify the public and private attributes associated with each sample in the test dataset. These attributes are denoted by $\hat{\tilde{y}}$ and $\hat{y}$ respectively. The classification model will be needed as the true public and private attributes associated with the input data are not known at anonymization time. The identification of public attribute is necessary to select a CVAE network for ObscureNet as discussed in the previous section.

Three different anonymization techniques can be implemented using ObscureNet. We refer to these techniques as *deterministic modification*, *probabilistic modification*, and *randomized* approach. They differ in whether they utilize the identified private attribute, and how they modify this attribute before it is used as a condition for the probabilistic decoder. We explain each of these approaches below.

4.2.1 Deterministic modification of the identified private attribute. This anonymization technique involves an injective function which deterministically maps each private attribute class in $\mathcal{A}$ to a different class in that set. This injective function is labelled as *private attribute modifier* in Figure 3. The identified private attribute (i.e., output of the classifier) is changed through the use of this modifier. We then pass the one-hot encoding of its output along with the latent representation of input data to the decoder which produces a new version of the input data, denoted by $\tilde{x}$.

4.2.2 Probabilistic modification of the identified private attribute. Probabilistic modification is similar to the deterministic one with one exception: the mapping of private attributes is done probabilistically. That is, one of the K private attribute classes, $a_1, \cdots, a_K$, is

Algorithm 2: Sensor Data Anonymization w/ ObscureNet

Data: Data embedding $\boldsymbol{x}$, autoencoder parameters θ, ϕ
Result: Anonymized embedding $\tilde{\boldsymbol{x}}$
$\hat{y}, \ \hat{\tilde{y}} \leftarrow \text{Classify}(\boldsymbol{x})$
$\boldsymbol{\mu}, \boldsymbol{\sigma} \leftarrow \text{Enc}_{\hat{\tilde{y}}}(\boldsymbol{x}; \theta)$
$\boldsymbol{\epsilon} \sim N(0, I)$
$\boldsymbol{z} \leftarrow \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$y' \leftarrow \text{Modify}(\hat{y})$ `// or` $y' \leftarrow$ `Randomize()`
$\tilde{\boldsymbol{x}} \leftarrow \text{Dec}_{\hat{\tilde{y}}}(\boldsymbol{z}, y'; \phi)$

picked at random for each sample in the test dataset and the decoder is fed the one-hot encoding of this private attribute class along with the latent representation of data. The probabilistic modification is an effective defence against the user re-identification attack as we discuss in Section 6.

4.2.3 Randomized approach. The third anonymization technique eliminates the need for a classification model to identify the private attribute. Rather than identifying the private attribute first and modifying its one-hot encoding, it simply passes a *stochastic vector* along with the latent representation of data to the decoder to produce a new version of this data. A stochastic vector is a vector of size K with non-negative entries that add up to 1. This technique aims to prevent user re-identification, but is simpler than the probabilistic modification technique as it does not require training an additional classification model for the private attribute.

Algorithm 2 shows the steps of the deterministic and probabilistic modification techniques. In the randomized approach, a randomly generated stochastic vector is used instead of the one-hot encoding of y'. In Section 6, we compare the three anonymization techniques presented above in terms of their ability to prevent user re-identification.

5 DATASETS

We use two open HAR datasets, namely MotionSense and MobiAct, to evaluate the efficacy of ObscureNet and other baseline methods from Section 3 in reducing the accuracy of intrusive inferences about private attributes of data while maintaining the accuracy of desired inferences about its public attributes. We describe these datasets below and elaborate on the process of creating embeddings of time series data generated by sensors.

5.1 MobiAct Dataset

The MobiAct [4] dataset is comprised of IMU readings from accelerometer and gyroscope sensors. The readings are collected from 66 subjects performing 12 different activities including walking, running, climbing up and down the stairs. We consider data from a group of 37 subjects only to create a more balanced dataset which has roughly the same number of male and female subjects. Out of the 37 subjects we select, 17 are female and the remaining 20 are male. Also from the 12 different activities captured in the dataset, we consider the following 4 activities: walking, standing, jogging, and climbing up the stairs. We choose these activities for two reasons. First, these are the same activities captured in the MotionSense dataset so we can make a comparison between the

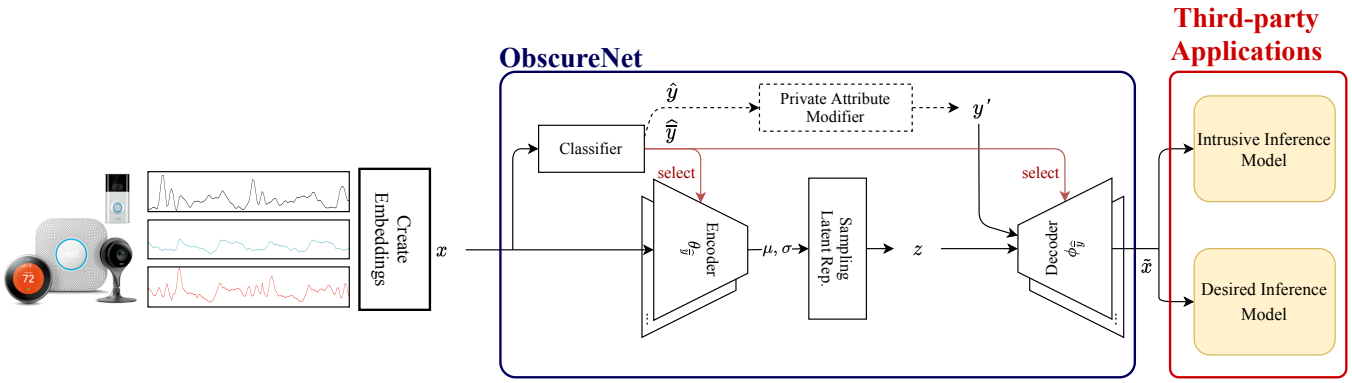

Figure 3: Anonymization with ObscureNet.

two datasets. Second, limiting our study to these activities partly addresses the class imbalance problem.

We assume in our experiments that the activity exercised by a subject is the public attribute and must be inferred by a fitness tracking application. However, the weight and gender of the subject are deemed private and their inference by this application or other third-party applications is regarded a violation of user privacy. We model gender as a binary attribute since these are the two classes that are present in our dataset. We model weight as a ternary attribute using a simple binning strategy which tries to assign roughly the same number of subjects to each bin. This helps to address the class imbalance problem. In particular, subjects who weigh less than or equal to 70 kg are assigned to weight-group 0. Subjects who weigh between 70 and 90 kg are assigned to group 1 and the rest are assigned to weight-group 2.

5.2 MotionSense Dataset

The MotionSense [24] dataset is collected by accelerometer and gyroscope sensors of an iPhone 6s using the Sensing Kit framework [15]. The data obtained from accelerometer and gyroscope has a sampling rate of 50 Hz. Each reading consists of 12 features, including attitude (roll, pitch, yaw), gravity, rotation rate, and user acceleration in three dimensions. As described in [24], data collection occurred when the smartphone was placed inside the pocket of the subjects wearing fit clothing.

MotionSense contains data from 24 subjects (14 male and 10 female subjects). Each individual in this dataset performs 15 trials of 6 different activities. These activities include climbing up and down the stairs, walking, jogging, sitting, and standing. The subjects in this dataset have a wide range of values for their age, weight, and height. Following [24] we combine the standing and sitting activities into one activity. This is done because it is difficult to tell apart these two activities using acceleration readings. Similar to the MobiAct dataset, we assume in our experiments that the activity exercised by a subject is the public attribute. We also assume that the gender identity of subjects is the private attribute and a third-party application should not be able to detect it.

From all the 15 trials, we use trials 11, 12, 13, 14, 15, and 16 to build our test set. This is similar to the test set in [24].

5.3 Embedding Sensor Data

It is shown in related work that activities can be identified more accurately if a number of consecutive IMU samples are analyzed at once. We call this an embedding of sensor data. To create our data embeddings, we use a window of size 128 samples. The window is moved with strides of 10 samples to create the next embedding. For the MotionSense dataset, we combine features along the three axes to create one feature (i.e., the magnitude). However, for the MobiAct dataset we use readings along the three axes as three separate features. Our experiments suggest that using three-dimensional sensor data increases the model accuracy.

We use a trial-based partitioning of training and test data for MotionSense, and a partitioning for MobiAct dataset with 80% training set and 20% test set.

6 EVALUATION

We implemented ObscureNet in Python using Keras and PyTorch libraries. It is an open-source package and can be downloaded from our GitHub repository[2]. The encoder and decoder are instantiated as neural networks with 5 layers and ReLU activation function in hidden layers. We set the number of latent variables to 10 and 5 in MotionSense and MobiAct datasets, respectively. The discriminator network is a multilayer perceptron model (MLP) with 4 layers.

We use Convolutional Neural Networks (CNNs) as sensitive and desired inference models (the right side of Figure 3). These models are used to assess data utility and privacy loss (i.e., accuracy of sensitive and desired inferences) before and after anonymization. We use max pooling and dropout layers to avoid overfitting.

Other baseline methods are implemented in Python using Keras and PyTorch libraries. In addition to these baselines, we compare against the state-of-the-art autoencoder-based anonymization techniques, i.e., Anonymization Autoencoder (AAE) [24]. We obtained their implementation from GitHub and ran it on both datasets.

6.1 Comparison with Baselines

We compare ObscureNet with the baseline anonymization methods discussed in Section 3. Unless otherwise stated, ObscureNet is

[2]https://github.com/sustainable-computing/ObscureNet

trained with hyperparameters that are set as follows: $\alpha = 0.2$ and $\beta = 2$. We consider the following baseline methods:

- 'General VAE' and 'Attribute-specific VAE' which rely on the mean manipulation technique explained in Section 3.1. The only distinction between these two methods is that the former, which is the method proposed in [11], uses a single VAE to anonymize all samples regardless of the value of their public attribute. The latter, however, trains different VAEs for different public attributes. At anonymization time, it first detects the public attribute of input data and then chooses the appropriate VAE for learning and manipulating the latent representation.
- 'Attribute-specific CAE' and 'Attribute-specific CVAE' are conditional generative models where the condition represents the private attribute. Data is anonymized by altering the condition variable before sending it to the decoder as discussed in Section 3.2. The difference between these two baselines is that in the former the condition is introduced in a vanilla autoencoder (resembling the architecture of Fader Networks which are developed for manipulating images [19]), whereas in the latter the condition is introduced in a variational autoencoder. Both methods train and utilize different autoencoders for different public attributes.
- 'Anonymization Autoencoder (AAE)' which is proposed in [24]. It does not uses conditional generative models. Instead it takes advantages of several regularizer models for adversarial training.

Moving from the top to the bottom of this list, the baseline methods combine different ideas to increase disentanglement of latent variables, making them more efficient and capable of concealing private attributes. Through ablations, we highlight the importance of incorporating each of these ideas in the design of ObscureNet, which essentially adds adversarial information factorization to the 'Attribute-specific CVAE' baseline and leverages non-deterministic private-attribute modifiers to prevent re-identification of private attributes after anonymization.

6.1.1 Accuracy of Sensitive and Desired Inferences. As the first step in our evaluation, we look at the accuracy of sensitive and desired inference models when their input is the original data and when it is the data anonymized by ObscureNet and other baselines. We evaluate these methods in three different anonymization tasks: gender anonymization in MotionSense, gender anonymization in MobiAct, and finally weight-group anonymization in MobiAct. Activity detection is the desired inference in all three tasks. We only study the problem of hiding a single private attribute. An extension to the case where there are multiple private attributes to be obscured simultaneously is discussed in Section 6.3.

The results reported for ObscureNet in this section are obtained using a deterministic private attribute modifier. We argue that if the deterministic modifier can reduce the accuracy of a sensitive inference to zero, through randomization, we can achieve the level of accuracy of a random guess. This also prevents re-identification of private attributes. Thus, we favour a lower accuracy for sensitive inferences.

Table 1 shows the inference accuracy achieved when using the output of different baselines and ObscureNet in the MotionSense gender anonymization task. Moreover, the overall F1-scores for activity and gender inferences are given in the last two columns. The first row, labelled 'Original Data', indicates the accuracy of activity and gender inference models on the original (unanonymized) data. It can be readily seen that using attribute-specific VAEs improves the F1-score of activity inference from 65.51% obtained by a General VAE to 72.45%. This can be attributed to the fact that having a specific VAE for each public attribute can partly address the imbalance problem in the training data[3]. Unfortunately this comes at the price of increasing the F1-score of gender inference. Comparing attribute-specific CAE and attribute-specific CVAE, we observe that both methods achieve comparable results for activity inference, but attribute-specific CVAE can effectively lower the gender inference accuracy and F1-score. We attribute this to the fact that variational autoencoders increase disentanglement of latent variables and allow for easy generalization compared to vanilla autoencoders. We witness that the four baseline methods we discussed so far either fail to obscure the private data or significantly reduce its usefulness for desired inferences.

The Replacement Autoencoder [25] cannot successfully obscure gender as the average accuracy results of activity and gender inferences are 96.3% and 97.1%, respectively. AAE [24] significantly reduces the gender inference accuracy, but cannot beat ObscureNet. ObscureNet reduces the F1-score of gender inference by an additional 36%. This is done while achieving a comparable activity inference F1-score with the best baseline methods. It is worth mentioning that going downstairs is the most difficult activity to detect after concealing the gender attribute. Comparing ObscureNet with the technique proposed in [35], which reduces the gender inference accuracy to roughly 60% as reported by the authors, we can conclude that our anonymization technique has better performance[4].

Next, we investigate gender anonymization results from the MobiAct dataset. Table 2 shows the accuracy of activity and gender inferences on the original data and the data anonymized by ObscureNet and baseline methods. The results are quite similar to the gender anonymization results from the MotionSense dataset. In this case going upstairs is the most difficult activity to detect after concealing the gender attribute. Compared to AAE, ObscureNet can significantly decrease the accuracy and F1-score of gender inference (by more than 30%) with a small loss of data utility ($\sim 6\%$). Compared to attribute-specific CVAEs which has the best performance among the baselines, ObscureNet can decrease the F1-score of gender inference by an additional 8%.

Lastly, we consider weight-group anonymization results from the MobiAct dataset. Recall that weight-group is a ternary private attribute; thus, this experiment is to check if ObscureNet can hide non-binary private attributes. It can be readily seen from Table 3 that ObscureNet outperforms all baselines in terms of the intrusive

[3]It also reduces the size of the model. To illustrate this, we calculated the total size of the general VAE and multiple VAEs models. The general VAE model, for the MotionSense dataset, has 24.5 million trainable parameters. In case of the MobiAct dataset, the general model has 8.7 million trainable parameters. Each of the attribute-specific VAEs has 1.7 million trainable parameters in case of the MobiAct dataset (a total of roughly 7 million trainable parameters for all VAEs) and 0.5 million trainable parameters in case of the MotionSense dataset (a total of 2 million trainable parameters for all VAEs).

[4]We were unable to reproduce the results of [35] and did not find their code in a public repository. Hence, we cannot use it as a baseline in the MobiAct dataset.

Table 1: Gender anonymization results from the MotionSense dataset. For each activity, the number of embeddings in the test set used for evaluation is shown in parentheses.

Method	Inference Accuracy								Accuracy/F1-score	
	Downstairs (1.9k)		Upstairs (2.5k)		Walking (6.2k)		Jogging (2.7k)		Overall	
	Activity	Gender	Activity	Gender	Activity	Gender	Activity	Gender	Activity	Gender
Original Data	95.58%	87.67%	93.19%	90.86%	98.71%	95.16%	97.28%	95.5%	96.93 / 95.90%	93.35 / 93.10%
General VAE	71.49%	38.62%	80.24%	28.34%	91.5%	45.12%	83.43%	38.61%	84.80 / 65.51%	39.72 / 38.05%
Attribute-specific VAEs	91.94%	76.22%	85.09%	64.05%	93.14%	77.9%	97.02%	23.12%	77.77 / 72.45%	63.94 / 61.95%
Attribute-specific CAEs	92.81%	63.74%	92.75%	77.60%	98.46%	76.13%	97.21%	85.52%	96.33 / 95.23%	76.54 / 74.81%
Attribute-specific CVAEs	92.09%	60.25%	93.31%	71.78%	96.13%	52.98%	96.8%	72.90%	95.39 / 94.05%	61.60 / 59.30%
AAE [24]	84.65%	57.91%	97.18%	57.64%	91.82%	52.89%	99.65%	46.23%	93.39 / 92.01%	53.15 / 42.83%
ObscureNet	87.52%	27.48%	92.83%	19.0%	98.71%	15.94%	96.87%	10.5%	95.63 / 94.23%	17.06 / 16.34%

Table 2: Gender anonymization results from the MobiAct dataset. For each activity, the number of embeddings in the test set used for evaluation is shown in parentheses.

Method	Inference Accuracy								Accuracy/F1-score	
	Walking (42.9k)		Standing (43.2k)		Jogging (4.2k)		Upstairs (1k)		Overall	
	Activity	Gender	Activity	Gender	Activity	Gender	Activity	Gender	Activity	Gender
Original Data	98.09%	99.63%	99.53%	95.52%	99.78%	99.28%	95.45%	94.47%	98.82 / 91.46%	97.61 / 97.52%
General VAE	92.34%	88.53%	98.56%	63.83%	90.52%	87.07%	52.93%	66.47%	94.77 / 77.55%	76.54 / 76.49%
Attribute-specific VAEs	95.48%	90.06%	99.63%	52.73%	98.14%	93.36%	93.32%	82.68%	97.54 / 86.23%	72.53 / 75.63%
Attribute-specific CAEs	93.22%	25.06%	99.59%	80.89%	97.47%	75.68%	94.82%	78.45%	96.45 / 83.15%	54.39 / 54.37%
Attribute-specific CVAEs	92.66%	14.65%	99.65%	28.75%	96.55%	25.67%	94.3%	58.81%	96.16 / 81.86%	22.31 / 22.35%
AAE [24]	96.96%	58.13%	99.61%	42.12%	99.61%	56.72%	84.44%	58.60%	98.19 / 87.98%	50.49 / 45.73%
ObscureNet	91.82%	5.39%	99.54%	23.48%	96.11%	10.55%	91.10%	24.46%	95.66 / 81.23%	14.39 / 14.28%

Table 3: Weight-group anonymization results from the MobiAct dataset. For each activity, the number of embeddings in the test set used for evaluation is shown in parentheses.

Method	Inference Accuracy								Accuracy/F1-score	
	Walking (42.9k)		Standing (43.2k)		Jogging (4.2k)		Upstairs (1k)		Overall	
	Activity	Weight	Activity	Weight	Activity	Weight	Activity	Weight	Activity	Weight
Original Data	98.17%	97.42%	99.55%	85.85%	99.74%	93.36%	95.16%	76.69%	98.86 / 91.96%	91.53 / 91.95%
General VAE	81.88%	46.86%	90.34%	44.59%	54.73%	56.13%	18.47%	47.15%	83.94 / 59.84%	46.22 / 37.14%
Attribute-specific VAEs	92.83%	70.04%	99.66%	71.79%	97.7%	53.87%	93.33%	61.34%	96.29 / 81.79%	70.03 / 65.51%
Attribute-specific CAEs	94.23%	49.75%	99.65%	76.85%	97.82%	88.21%	94.87%	59.79%	96.97 / 84.22%	64.45 / 64.21%
Attribute-specific CVAEs	94.88%	26.59%	99.7%	19.37%	94.59%	53.55%	95.44%	51.41%	97.15 / 84.28%	24.69 / 21.46%
AAE [24]	97.39%	63.77%	99.35%	50.15%	98.91%	72.66%	90.91%	57.16%	98.32 / 88.50%	57.66 / 56.44%
ObscureNet	94.22%	7.58%	99.59%	14.12%	96.40%	21.07%	91.37%	29.5%	96.83 / 83.40%	11.54 / 10.80%

inference accuracy by a huge margin. This is while it only reduces the data utility slightly, i.e., by less than $\sim 6\%$ compared to AAE.

To conclude, our experiments show that ObscureNet outperforms the baselines and autoencoder-based anonymization techniques from related work in all three tasks we studied in this section. In particular, it obscures gender in the MotionSense dataset while yielding the same utility as the best baseline. In the MobiAct dataset, it completely obscures gender and weight-group with a small loss of data utility. We believe that data utility can be further improved by using different hyperparameters as discussed in Section 6.2. In the next section, we examine the effects of different private-attribute modifiers and corroborate that ObscureNet can prevent re-identification of private attributes thanks to non-deterministic modifications of these attributes. This is a major improvement over other autoencoder-based anonymization techniques [24, 25].

6.1.2 Non-deterministic Anonymization. In this section, we compare the three anonymization techniques which can be implemented using ObscureNet and were described in Section 4.2. Two of the three techniques, namely probabilistic modification and randomized approach, add randomness to the anonymization process. This can effectively prevent an adversary from passing a dataset with known private attributes through ObscureNet and training a model to recover the original data based on the anonymized data and true private attributes.

Table 4 shows the accuracy of desired and sensitive inferences when the private attribute (noted in parentheses) is obscured using ObscureNet. Expectedly, the deterministic modifier yields the lowest sensitive inference accuracy because, unlike the other two techniques, it modifies the private attribute at all times. However, as we discuss in the next section, private attributes can be easily re-identified due to the deterministic nature of this anonymization. Results for the other two techniques are quite similar; they can reduce the intrusive inference accuracy to the level of a random guess. A nice property of the randomized approach is that it does not need to use a model to detect the private attribute before modifying it. This makes it a suitable and more practical choice for anonymization on resource-constrained devices.

6.1.3 Re-identification Accuracy. In the previous section, the ability of ObscureNet to anonymize private data was evaluated in deterministic and non-deterministic cases. Although the deterministic private-attribute modifier does a better job of reducing the accuracy of a sensitive inference, we show here that it cannot prevent the re-identification attack.

The re-identification attack exploits the deterministic nature of data anonymization to foil the anonymization process [11]. Suppose 20% of the anonymized data is leaked to the attacker, i.e., they know the true private attribute associated with this data and can leverage this knowledge to train a model to re-identify the true attribute of the rest of data. To get this 20%, we randomly choose 20% of the anonymized data and evaluate the accuracy of the re-identification attack. We do 20 independent runs and report the average and standard deviation of the accuracy of the re-identification model. Figure 4 illustrates that both ObscureNet and Anonymization Autoencoder [24] fail to completely prevent the re-identification attack due to the deterministic nature of anonymization they perform. However, probabilistic modification and randomized approach can both greatly reduce the accuracy of a re-identification model.

Figure 4 shows that the randomized approach, which is easier to deploy than the probabilistic attribute modifier, has nearly the same performance as the probabilistic one in the gender anonymization task. But, in the weight-group anonymization task, it further reduces the re-identification accuracy by roughly 13%.

6.2 Investigating Utility-Privacy Trade-offs

In this section, we investigate how the anonymization performance of ObscureNet can be enhanced by adjusting the two hyperparameters, α and β, when networks are being trained. Furthermore. we explore if users can trade utility for privacy by adjusting the hyperparameters. For brevity, we only study the gender anonymization problem using the MotionSense dataset and report the results when the deterministic attribute modifier is adopted. Since we neglect the possibility of user re-identification in this stage, the best anonymization technique would be the one that reduces the accuracy of the sensitive inference to zero.

Recall the adversarial loss function of ObscureNet expressed in Equation (4). Intuitively, higher α encourages information factorization, which subsequently prevents the leak of private information through the latent representation. But it can lower the reconstruction quality because VAE's ELBO gets a lower relative importance.

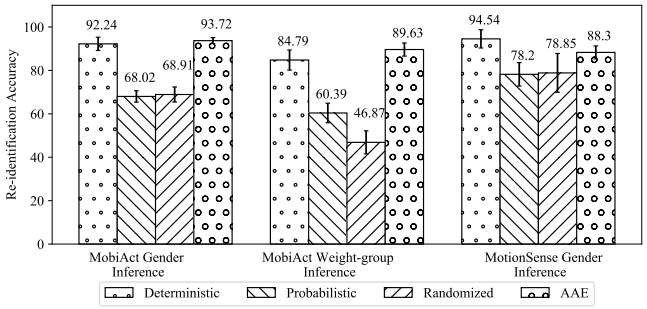

Figure 4: Comparing the inference accuracy of the re-identification model averaged over 20 runs on the sensor data anonymized by ObscureNet using different attribute modifiers. Error bars show 2σ from the mean.

Similarly, higher β encourages the disentanglement of latent variables but reduces the reconstruction quality. Thus, it is possible to achieve different utility-privacy trade-offs by tuning α and β.

Figures 5 and 6 show respectively the accuracy of desired and sensitive inferences and how it changes with α and β values. We consider 6 values of β and 4 values of α; these values are intentionally chosen from the logarithmic scale to examine the range of behaviour we can expect from ObscureNet. We assume β can take values from $\{0.1, 0.2, 0.5, 1, 2, 10\}$ and α can take values from $\{0.1, 0.2, 1, 10\}$.

As it can be seen from Figure 5, for a fixed value of α, shifting β to the two extremes, i.e., 10 or 0.1, would diminish the utility of data, although the decline in utility is more pronounced when $\beta = 10$. Another observation is that for a fixed β, the value of α does not seem to drastically affect the utility of data unless it is equal to 10. We attribute this to the fact that when $\alpha = 10$, information factorization overwhelms the VAE's reconstruction loss.

In Figure 6, we can see that increasing the value of β from 0.2 to 2 lowers the accuracy of the sensitive inference in general. However, moving β to any of the two extremes diminishes the anonymization performance of the ObscureNet for all α values. Comparing the curves for different values of α suggests that $\alpha = 0.1$ is almost always better than other values of α regardless of the value of β.

Considering the accuracy of both desired and sensitive inferences, it turns out setting α to 0.2 and changing β between 0.1 and 2 yields the Pareto frontier. For example, the user can trade utility for privacy by setting β to 2 and do the opposite by setting β to 0.1. While we only tried a small number of choices for α and β, we already showed that it is possible to navigate the privacy-utility trade-off by tuning these weights.

6.3 Obscuring Multiple Private Attributes

We now turn our attention to the case where there are multiple private attributes. Specifically, we treat gender and weight-group of subjects in the MobiAct dataset as private attributes and consider their activity as the public attribute. We evaluate the anonymization performance of a single ObscureNet model, which can hide both private attributes at the same time, with deterministic, probabilistic, and randomized modifiers. In this case, ObscureNet conditions

Table 4: Accuracy of sensitive and desired inferences using ObscureNet with different private-attribute modifiers in the three anonymization tasks.

	Original data		Random guess	Deterministic		Probabilistic		Randomized	
	Activity	Private	Private	Activity	Private	Activity	Private	Activity	Private
MotionSense (Gender)	96.94	93.33	50.00	95.61	16.95	96.02	54.32	95.99	58.49
MobiAct (Gender)	98.82	97.52	50.00	95.72	14.43	97.02	52.70	96.26	54.20
MobiAct (Weight-group)	98.84	91.67	33.33	96.86	11.47	97.57	49.78	97.45	43.02

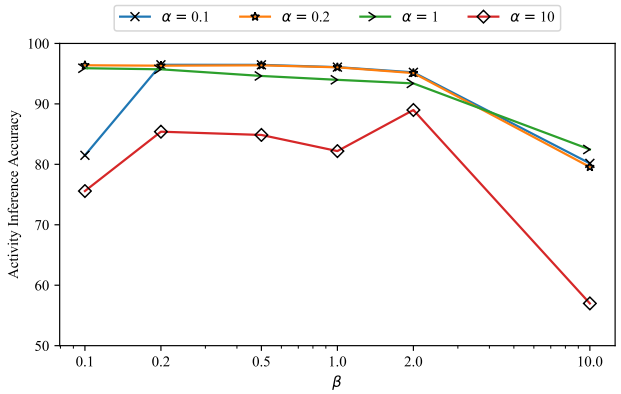

Figure 5: The accuracy of activity inference with varying α and β values. Note that the x-axis has logarithmic scale and the y-axis is exaggerated.

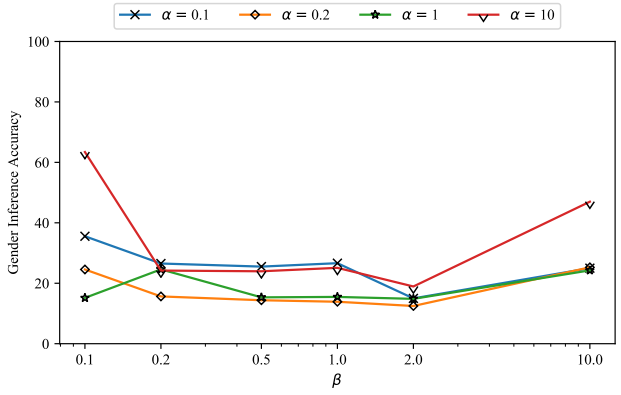

Figure 6: The accuracy of gender inference with varying α and β values. Note that the x-axis has logarithmic scale.

the probabilistic decoder on both private attributes and uses two discriminator networks, one for each private attribute. We train ObscureNet in adversarial setting as described in Section 4.1.

This problem is interesting because in practice the user often wishes to anonymize multiple private attributes simultaneously. Figure 7 shows the result of joint anonymization of gender and weight-group attributes, while preserving information about the activity in the anonymized data. It is evident that ObscureNet with

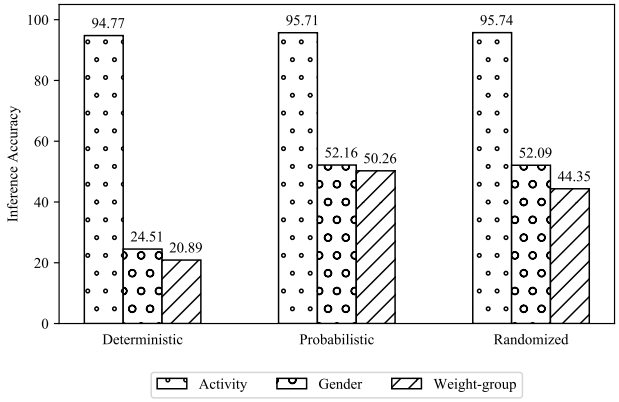

Figure 7: Inference accuracy of ObscureNet with different attribute modifiers when it is trained to hide two private attributes at the same time.

a deterministic attribute modifier can successfully reduce the accuracy of both sensitive inferences to less than 25% while achieving the accuracy of 94.8% for the desired inference. Should we use the probabilistic modifier or the randomized approach, the accuracy of both sensitive inferences would get close to the level of a random guess. This indicates that a single ObscureNet model can effectively hide multiple private attributes.

Nevertheless, comparing this result with the result of removing each private attribute using a separate ObscureNet model (cf. Table 2 and Table 3), we can see that the anonymization performance is slightly degraded. Therefore, if the execution time and required resources are not an issue, an alternative approach for anonymizing several attributes could be to create an anonymization pipeline by utilizing multiple ObscureNet models (each concealing only one private attribute) and feeding the output of one model to the next model in the pipeline.

6.4 Deployment on IoT Devices

We finally investigate if ObscureNet can run on a Raspberry Pi 3 Model B to anonymize sensor data in real time. We use the Raspberry Pi as an IoT device that collects data from several sensors and runs ObscureNet locally to anonymize the collected data. We install Keras and PyTorch libraries on the Raspberry Pi, and report the running time of ObscureNet when the private attribute is modified in a probabilistic fashion[5]. We assume the encoder and decoder

[5]The running time would be even lower if we adopt the randomized approach. This is because we do not need to predict the private attribute before modifying it.

Table 5: The running time of ObscureNet when anonymizing one embedding in different tasks

Anonymization task	Running time (ms)/Embedding				Total running time (ms) per embedding
	Desired Inference	Sensitive Inference	Probabilistic Encoder	Probabilistic Decoder	
MotionSense (Gender)	0.60	0.62	0.86	0.83	2.91
MobiAct (Gender)	10.39	10.14	9.33	1.15	31.01
MobiAct (Weight-group)	10.05	10.05	9.78	1.18	31.06

networks of ObscureNet are trained in a server, where the training data resides, and the weights are sent to the IoT device prior to anonymization.

To make real-time anonymization possible on the the Raspberry Pi, ObscureNet must be able to anonymize an embedding before the next one becomes available. Recall that in both datasets, we set the stride length to 10 samples, which means that the next embedding is created after receiving 10 sensor readings. The sampling rate of the IMU sensor is respectively 50Hz and 20Hz in MotionSense and MobiAct. Hence, a new embedding is created every 200 milliseconds in MotionSense, and every 500 milliseconds in MobiAct. If the running time of ObscureNet per embedding is less than this, it will be able to perform anonymization in real time.

As illustrated in Figure 3, an execution of ObscureNet can be divided into four main steps: (1) predicting the public and private attributes associated with the original data using the pre-trained desired and sensitive inference models, (2) encoding the input data through an attribute-specific encoder, (3) modifying the predicted private attribute, and (4) decoding the latent representation together with the modified attribute through an attribute-specific decoder. The first step involves running both inference models. The second and fourth steps require selecting the attribute-specific encoder and decoder networks according to the predicted public attribute. The third step involves generating a random number to determine how the private attribute should be modified. We ignore the running time of this step as it is negligible compared to the running time of the other three steps.

Table 5 shows the running time (in milliseconds) of the main steps in ObscureNet in three different anonymization tasks, namely gender anonymization in MotionSense, and gender and weight-group anonymization in MobiAct. To obtain the running time per embedding, we calculated the total running time of each step for approximately 8,000 embeddings and then divided this by the number of embeddings. Note that the running times of the attribute-specific encoder and decoder networks depend on the predicted public attribute. In this table, we only report the worst-case running times of the attribute-specific encoder and decoder networks across different activities (i.e., values of the public attribute).

Considering the gender anonymization task in MobiAct, the activity and gender inference models take roughly 10 milliseconds each to predict the private and public attributes of one embedding. The encoder and decoder running times for one input data embedding are around 9 and 1 milliseconds, respectively. These add up to 31 milliseconds per embedding. In the case of weight-group anonymization, the running times also add up to roughly 31 milliseconds. Given the time budget of 500 milliseconds, our results show that ObscureNet can anonymize the gender and weight-group attributes of participants in the MobiAct dataset in real time on a Raspberry Pi 3 model B.

Turning our attention to the gender anonymization task in MotionSense, we find that predicting the private and public attributes takes much less time. In particular, gender and activity inferences complete in 0.62 and 0.60 milliseconds, respectively. Moreover, the encoder and decoder networks take respectively 0.86 and 0.83 milliseconds to run. Thus, the total running time of ObscureNet would be 3 milliseconds per embedding. Given the time budget of 200 milliseconds, we corroborate that ObscureNet is capable of anonymizing input data embeddings of MotionSense in real time.

7 CONCLUSION

With the rapid adoption of consumer IoT devices, from indoor flying drones to robot vacuums and smart thermostats, a plethora of sensors will soon be installed in our homes and workplaces. Since the data gathered by these sensors can enable many useful services, it is anticipated that the existing all-or-nothing models for data sharing will force many users to give up on their privacy to benefit from these services. Despite the significant amount of research that has been done in recent years on understanding and addressing trade-offs between utility and privacy, there is still no anonymization technique that offers acceptable levels of utility and privacy loss, while preventing user re-identification. ObscureNet addresses this gap in the literature through the use of conditional generative models and information factorization. To our knowledge, these ideas have not been previously applied to the sensor data anonymization problem.

Our experiments on two HAR datasets suggested that ObscureNet can reduce the accuracy of intrusive inferences by an additional 13.48% on average compared to the best autoencoder-based baseline without causing a significant drop in the accuracy of desired inferences. We showed that ObscureNet can conceal multiple private attributes simultaneously and discussed how tuning its hyper-parameters enables users to navigate trade-offs between utility and privacy. We believe this is an important property of our anonymization technique as users of IoT devices naturally have different expectations and concerns about applications they install which work on their data. Furthermore, we confirmed that ObscureNet can run locally on an IoT device to anonymize sensor data in real time.

In future work, we plan to use ObscureNet to remove private attributes from the data collected from indoor environment. We aim to provide users with abstract ways to describe their privacy concerns and add a slider knob to the anonymization technique that can be used to indicate how important a privacy threat is to them or how much they are willing to sacrifice their privacy in return for higher utility. Lastly, in this paper we assumed that a public dataset is available for training ObscureNet. We will relax this assumption in future work and investigate how ObscureNet can be trained in a federated learning setting.

REFERENCES

[1] Amr Alanwar, Yasser Shoukry, Supriyo Chakraborty, Paul Martin, Paulo Tabuada, and Mani Srivastava. 2017. PrOLoc: Resilient localization with private observers using partial homomorphic encryption. In *2017 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 41–52.

[2] Fayez Alharbi, Lahcen Ouarbya, and Jamie A Ward. 2020. Synthetic Sensor Data for Human Activity Recognition. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–9.

[3] Ranya Aloufi, Hamed Haddadi, and David Boyle. 2020. Privacy-preserving Voice Analysis via Disentangled Representations. *arXiv preprint arXiv:2007.15064* (2020).

[4] Charikleia Chatzaki, Matthew Pediaditis, George Vavoulas, and Manolis Tsiknakis. 2017. Human Daily Activity and Fall Recognition Using a Smartphone's Acceleration Sensor. In *Information and Communication Technologies for Ageing Well and e-Health*. Springer, 100–118.

[5] Dong Chen, David Irwin, Prashant Shenoy, and Jeannie Albrecht. 2014. Combined heat and privacy: Preventing occupancy detection from smart meters. In *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 208–215.

[6] Antonia Creswell, Anil A. Bharath, and Biswa Sengupta. 2017. Conditional Autoencoders with Adversarial Information Factorization. *CoRR* abs/1711.05175 (2017). arXiv:1711.05175

[7] Dan Feldman, Chongyuan Xiang, Ruihao Zhu, and Daniela Rus. 2017. Coresets for differentially private k-means clustering and applications to privacy in mobile sensor networks. In *2017 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 3–16.

[8] Clément Feutry, Pablo Piantanida, Yoshua Bengio, and Pierre Duhamel. 2018. Learning anonymized representations with adversarial neural networks. *arXiv preprint arXiv:1802.09386* (2018).

[9] Jonathan Francis et al. 2019. OccuTherm: Occupant Thermal Comfort Inference Using Body Shape Information. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. ACM, 81–90.

[10] Ian Goodfellow. 2016. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160* (2016).

[11] Omid Hajihassani, Omid Ardakanian, and Hamzeh Khazaei. 2020. Latent Representation Learning and Manipulation for Privacy-Preserving Sensor Data Analytics. In *IEEE Second Workshop on Machine Learning on Edge in Sensor Systems (SenSys-ML)*. 7–12.

[12] Nazia Hossain and Mahmuda Naznin. 2018. Sensing Emotion from Voice Jitter. *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems* (2018).

[13] Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. 2017. Context-aware generative adversarial privacy. *Entropy* 19, 12 (2017), 656.

[14] Jeya Vikranth Jeyakumar, Liangzhen Lai, Naveen Suda, and Mani Srivastava. 2019. SenseHAR: a robust virtual activity sensor for smartphones and wearables. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 15–28.

[15] Kleomenis Katevas, Hamed Haddadi, and Laurissa Tokarchuk. 2016. Sensingkit: Evaluating the sensor power consumption in iOS devices. In *12th International Conference on Intelligent Environments*. IEEE, 222–225.

[16] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*. 3581–3589.

[17] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations ICLR*.

[18] Jack Klys, Jake Snell, and Richard Zemel. 2018. Learning latent subspaces in variational autoencoders. In *Advances in Neural Information Processing Systems (NIPS'18)*. 6444–6454.

[19] Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Fader networks: Manipulating images by sliding attributes. In *Advances in neural information processing systems*. 5967–5976.

[20] Matthew Lentz, Rijurekha Sen, Peter Druschel, and Bobby Bhattacharjee. 2018. Secloak: Arm trustzone-based mobile peripheral control. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. 1–13.

[21] Xi'ang Li, Jinqi Luo, and Rabih Younes. 2020. ActivityGAN: generative adversarial networks for data augmentation in sensor-based human activity recognition. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*. 249–254.

[22] Zinan Lin, Alankar Jain, Chen Wang, Giulia Fanti, and Vyas Sekar. 2020. Using GANs for Sharing Networked Time Series Data: Challenges, Initial Promise, and Open Questions. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 464–483. https://doi.org/10.1145/3419394.3423643

[23] Sicong Liu, Junzhao Du, Anshumali Shrivastava, and Lin Zhong. 2019. Privacy adversarial network: representation learning for mobile data privacy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 4 (2019), 1–18.

[24] Mohammad Malekzadeh, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. 2019. Mobile sensor data anonymization. In *Proceedings of the International Conference on Internet of Things Design and Implementation*. 49–58.

[25] Mohammad Malekzadeh, Richard G Clegg, and Hamed Haddadi. 2018. Replacement AutoEncoder: A Privacy-Preserving Algorithm for Sensory Data Analysis. In *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation*. 165–176.

[26] Fan Mo, Ali Shahin Shamsabadi, Kleomenis Katevas, Soteris Demetriou, Ilias Leontiadis, Andrea Cavallaro, and Hamed Haddadi. 2020. DarkneTZ: Towards Model Privacy at the Edge Using Trusted Execution Environments. (2020), 161–174.

[27] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *ICML*.

[28] Chaoyue Niu, Fan Wu, Shaojie Tang, Lifeng Hua, Rongfei Jia, Chengfei Lv, Zhihua Wu, and Guihai Chen. 2020. Billion-scale federated learning on mobile clients: a submodel design with tunable privacy. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.

[29] Rayman Preet Singh, Benjamin Cassell, Srinivasan Keshav, and Tim Brecht. 2018. TussleOS: Managing privacy versus functionality trade-offs on IoT devices. *ACM SIGCOMM Computer Communication Review* 46, 3 (2018), 1–8.

[30] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*. 3483–3491.

[31] Jordi Soria-Comas, Josep Domingo-Ferrer, David Sánchez, and Sergio Martínez. 2014. Enhancing Data Utility in Differential Privacy via Microaggregation-Based k-Anonymity. *The VLDB Journal* 23, 5 (Oct. 2014), 771–794.

[32] Animesh Srivastava, Puneet Jain, Soteris Demetriou, Landon P Cox, and Kyu-Han Kim. 2017. CamForensics: Understanding visual privacy leaks in the wild. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. 1–13.

[33] Jiwei Wang, Yiqiang Chen, Yang Gu, Yunlong Xiao, and Haonan Pan. 2018. SensoryGANs: an effective generative adversarial framework for sensor-based human activity recognition. In *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[34] W. Xue, C. Luo, G. Lan, R. Rana, W. Hu, and A. Seneviratne. 2017. Kryptein: A Compressive-Sensing-Based Encryption Scheme for the Internet of Things. In *2017 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 169–180.

[35] Dalin Zhang, Lina Yao, Kaixuan Chen, Guodong Long, and Sen Wang. 2019. Collective protection: Preventing sensitive inferences via integrative transformation. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1498–1503.

[36] Mingmin Zhao, Fadel Adib, and Dina Katabi. 2016. Emotion Recognition Using Wireless Signals. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 95–108.

[37] Shilin Zhu, Chi Zhang, and Xinyu Zhang. 2017. Automating visual privacy protection using a smart LED. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. 329–342.