Analyzing Privacy of Time Series Data Using Substitute Autoencoder Neural Network

Sayantica Pattanayak
Computer Science
North Dakota State University
Fargo, USA
sayantica.pattanayak@ndsu.edu

Simone A. Ludwig
Computer Science
North Dakota State University
Fargo, USA
simone.ludwig@ndsu.edu

***Abstract*—Nowadays people are very concerned about their health. For example, people want to keep track of how often they are active throughout the day. Hence many smart devices are equipped with sensors which help to keep track of the activities. Users who use these sensors want to share the information to be analyzed for their own health benefits. However, at the same time the users do not want to share the sensitive information. In this paper, we propose a privacy preserving predictive model. We used the concept of denoising autoencoder to hide the sensitive attributes of a user. We divided the data sets into three different subsets: desired, sensitive, and non sensitive subsets. The output of the denoising autoencoder will only be the desired and non-sensitive subsets. The sensitive subsets are hidden by the non-sensitive subsets. We evaluated the efficacy of our predictive model using three different flavors of autoencoders (CNN, Deep, and LSTM). To retain the accuracy as similar as possible compared to the original data set, we used a convolutional neural network for classification. We measured the F1-score of our model against each of the three autoencoders.**

***Index Terms*—AutoEncoder Neural Networks, Generative Adversarial Neural Networks, Long Term Memory Neural Network, Convolutional Neural Network.**

I. Introduction

In recent years smart devices are used by almost every person. Smart devices initially were used for communication. People used it to access emails or social media applications. Slowly smart devices started implementing sensors, which collect data about a particular user [1]. The collection of data has both advantages as well as disadvantages. For example, GPS enabled smart phones tell the whereabouts of a person throughout the day, thus, invading a person's privacy [2]. On the other hand these smart devices are equipped with sensors, which can track a user's activities throughout the day. For example jogging, sitting, walking, running , drinking, smoking, and sleeping activities can be tracked. If a person does not meet his / her target, the sensor alerts the person, so that the person can be notified to meet the target. This information is sent to a cloud service to actuate a response. As the information benefits the health, the users want to share the data for analyzing.

Recently Wyze Labs [3], which makes smart cameras and connected home gadgets, has confirmed that the databases holding millions of customers' information were exposed to the public. Since data breach is increasing, privacy concerns have arisen from the customers. The customers do not want to share sensitive information with a third party. For example, a user wants to share the jogging and walking information but not the drinking or the smoking information. People nowadays do not trust the cloud services. Therefore, the main challenge today is how to sent the desired statistics to the cloud without interfering with personal information.

Moreover, the data collected from the smart devices is mostly time series data, so it is very challenging not to sent a part of the data to the cloud. We can only sent the data by transforming the time series data. By transforming we mean changing or perturbing the sensitive part of the time series data. However, we have to make sure that the transformed data should provide the same accuracy as the original data set. This is done to make sure that the transformed data set is not a misleading data set.

In this paper, we consider two time series data sets [4]. We then transform the time series data sets to be used by the cloud service. Motivated by the recent advancement in autoencoder, we implemented denoising autoencoder. In this paper we will show how a user will send only desired information to the cloud service for further research. This is achieved by the user trusting a trusted authority. The trusted authority collects the original data set and then transforms the data set. The transformed data set is used as the input to the denoising autoencoder to produce a transformed output. The transformed output has the sensitive attributes hidden. Instead of only using external noise to hide sensitive information, we will also use non-sensitive information from our data set to replace the sensitive information. To measure the efficacy of our model we will use different flavors of autoencoders: CNN, LSTM and Multilayer. We will also use a CNN as a classifier to check the accuracy of both the original and the transformed data set.

This paper is outlined as follows. In the first part of this paper, we describe the related work and the preliminaries required for understanding our research work. In the second part of the paper we describe our proposed model. In the third section we describe the experimental setting and results of our two data sets. The experimental setting gives a brief introduction to the time series data sets that are used. Afterwards,

978-1-7281-2547-3/20/$31.00 ©2020 IEEE

we describe the security part of our model followed by the conclusion.

II. Related Work

In this section we describe the various approaches related to our privacy preserving model. The various approaches, which are described here are the differential privacy method [5], the filtering method, data mapping method, and finally the substitution method.

Differential Privacy (DP) [6] is an approach, which adds constraints on the algorithm used to publish aggregate information. DP limits the disclosure of private information of records in the data sets. Furthermore, DP permits companies to access a large number of sensitive data for research analysis and business ananalysis without privacy breach. Also, research institutions use differential privacy technology to automate privacy processes within cloud-sharing communities across countries. For example, Apple uses DP to protect the privacy of users and resolve data sharing problem. The authors in [7] have shown the amount of noise to be added in differential privacy to make data secure. The authors have shown that by adding gaussian noise instead of laplacian noise will increase the computational efficiency. Also, the authors in [8] used differential privacy with machine learning. The authors used two standard public data sets, MNIST [9] and CIFAR-10 [10]. They improved the privacy of the data sets by introducing the Stochastic Gradient Descent (SGD) [11] algorithm. The major disadvantage of differential privacy is to add more noise if more data has to be hidden. This decreases the efficiency of a data set.

Another approach to maintain privacy in the data is by filtering. Instead of adding constraints on the algorithm, the filtering technique filters the sensitive information. Collaborative filtering is a very secure way to filter the sensitive information. This type of filtering has been best used for giving recommendation to the users. In [12], the authors used collaborative filtering for privacy preserving. They aimed at solving this problem by the systematic collection of sensitive information of preferences. They partitioned the data between parties to ensure privacy. Different techniques of collaborative filtering are meintioned in [13]. The paper gives an overview of the model-based, memory-based and hybrid-based collaborative filtering techniques. The key disadvantage of collaborative filtering is that it does not work well with sparse data sets [14].

Data Mapping is another technique to hide data. Data Mapping is a process to map data fields from the source fields to the target fields. One of the data mapping techniques used in healthcare privacy is geomasking [15]. The authors provide protection for individual addresses while maintaining spatial resolution for mapping purposes using geomasking. In [16], a new adaptive geomasking technique known as donut geomasking was proposed. The new technique extends the current method of geomasking by ensuring a user defined minimum level of geoprivacy. The authors in [17] propose another geographic masking technique known as location swapping. When locations of individual-level health data are released in the form of published maps, the identity of these individuals could be identified through reverse geocoding. Location swapping replaces an original location with a masked location selected from all possible locations with similar geographic characteristics within a specified neighborhood. The main disadvantage of this method that it is impossible to say that the feature extracted from the raw data set does not contain any sensitive information.

Another way of perturbing data is known as randomization of data. Randomization of data is a process of making data random. This could be done by generating a random permutation of a sequence, generating a random numbers, or by selecting random samples of the population. The authors in [18] proposed an approach whereby many clients can protect their personal information in a server. The clients can use a randomization algorithm to randomize the data and then sent it to the server. The authors have chosen the randomization technique so that the aggregate properties can be recovered with sufficient precision. The authors in [19] introduce a family of geometric data transformation methods (GDTMs) that distort confidential numerical attributes in order to meet privacy protection in clustering analysis.

In [20], the authors used symmetric key encryption to randomize the password. Different classifiers were used to analyze the accuracy of the model. Also, the authors in [21] randomized the sensitive attributes of a patient data set using fuzzy membership [22] functions to hide the sensitive data. However, these randomization methods have not been applied to time series data.

The data sets used in all the above mentioned related work are not time series data sets. The authors in [23] introduced the concept of replacement technique with time series datasets. Data replacement or data substitution is a technique that can be used with time series data. The technique replaces the sensitive data of a user by non-sensitive data. Then, the transformed data is sent to the cloud. The authors first divided the data sets into three subsets: desired, sensitive, and nonsensitive. Then, the authors replaced the sensitive attributes with the non-sensitive attributes. The authors applied the denoising autoencoder to transformed the data set. However, the authors limited their approach to only multilayer autoencoder. Thus, in our paper we are experimenting with different time series datasets and with several different autoencoder neural networks. We show and compare the efficiency of different flavors of autoencoder.

III. Preliminaries

In this section, we give a brief overview of the background information of our privacy preserving model. Our privacy preserving model uses two different deep learning neural networks: AutoEncoder and 1-D Convolutional Neural Network. Each of the neural networks is summarized below:

- Autoencoder: Autoencoder [24] is a data compression algorithm. An autoencoder is built based on two important functions: Encoding and Decoding. The encoding function compresses the feature vector to the most important feature vector. On the other hand, the decoding function decompresses the data to the original feature vector. The optimization of the encoding and decoding functions are done using Stochastic Gradient to minimize the reconstruction loss. The two functions in the autoencoder are very data specific and lossy. By data specific we mean that the autoencoder will only able to compress data on which they are trained on. By lossy we mean that the output of the decoder will be degraded compared to the original output.

- Denoising Autoencoder: One of the interesting practical applications of the autoencoder is data denoising. Autoencoders with more hidden nodes than the inputs learns the risk of learning identity functions. Identity functions return the same output as its input. If an autoencoder returns the same output as its input then it is of no use. Denoising autoencoders [25] addresses the problem of identification risk by purposefully corrupting (adding noise) the inputs. In general, the amount of nodes to be set to zero is about 50%. Others suggest a lower count of 30%. However, it depends on the amount of input nodes and data. If the data set is small we might have to add a higher percentage. The denoising autoencoder is used in our model to replace the sensitive attributes. We used different flavors of autoencoders incorporated with the denoising property to increase the efficacy of our model. The denoising property solves the problem of 'overfitting' by corrupting the data by randomly turning some of the input values to zero.
- Convolutional Neural Network: Convolutional Neural Network (ConvNets or CNN) are used in image recognition [26] and classification [27]. There are four operations performed by a CNN: convolutional step, nonlinearity, pooling, and classification. In the convolution step, the feature map is created from the data sets. After the convolutional step, ReLu is used as an non-linear operation. ReLu functions add non-linearity to the CNN since most of the data sets in the real world are nonlinear. Then pooling is a applied. Pooling is also known as subsampling or downsampling. This function reduces the dimentionality of the feature map. At the same time it retains the most important information. Along with the AutoEncoder we used a Convolutional Neural Network model to check if the efficiency of the raw and modified data is equal. Otherwise there is no use of providing a misleading data set.

IV. Proposed Approach

This section shows the predictive model to substitute the sensitive attributes of the data sets with nonsensitive attributes.

The modified data set can be used by a cloud server. Our proposed approach consists of three parties: user, trusted authority, and server.

To accomplish the privacy policy of the user we divided the data sets. Our data sets have been grouped into three subsets: black listed, white listed, and gray listed. The black listed subset consists of sensitive attributes, which the user does not want to reveal. The white listed subset consists of the desired information about the user. This information can benefit the user when shared. The gray listed subset consists of non-sensitive information. The user does not worry about the information, if it is shared. The tasks of each of the three parties are explained below:

- User: The user/subject from the skoda data set wore a sensor on the left hand as well as on the right hand. The data consists of classes 'write on notepad', 'open hood', 'close hood', 'check gap on the front door', 'open left front door', 'close left front door', 'check trunk gaps', 'open and close trunk', and 'check steering wheel'. The Hand gesture data set consists of two subjects. Each subject performed a certain activity. The activities/classes are 'open window', 'close window', 'water a plant', 'turn a book', 'drink a bottle', 'cut with knife', 'chop with knife', 'stir in bowl', 'forehand', 'backhand', 'smash'. The sensors collect time series data. The user wants to receive benefits from the data by sharing this information on the cloud. However, at the same time the user does not want to share the sensitive attributes such as 'write on notepad', 'open and close hood', 'drink a bottle', and 'turn a book'.
- Trusted Authority: The user trusts only the trusted authority. The trusted authority is a machine learning platform. The user sends the original time series data set to the Trusted Authority. Along with the data set, the user also sends the list of the three subsets. The three subsets consist of the desired, sensitive, and non-sensitive information provided by the user. Inspired by the denoising autoencoder, we implemented the model in our proposed approach. First, we substitute the black listed subset with the gray listed subset. To add more privacy we generated a noisy gaussian digit and clip the images between 0 and 1. Then, we added it to the transformed black listed subset. The transformed training set consists now of the original white listed subset, the original grey listed subset, and the transformed black listed subset. As we substitute the sensitive information with non-sensitive information, we call our autoencoder as substitute autoencoder. Then, we train the substitute autoencoder to map the transformed data set to the original data set.
- Server: The server is a third party who needs the desired information for further analysis. The user requests service from the server. The server allows the user to upload the time series data after sharing some information. The user does not trust the server, thus the user does not give the original data set to the server. The user gives the original data set to the trusted authority. The trusted authority is trusted by both the server and the user. The trusted authority transforms the data set by substituting the black subset with the grey subset. The transformed data set is then given to the server.

V. Experimental Setup

A. Data Set

Experiments are conducted on two data sets. The data sets are the Skoda [28] and the Hand gesture data set [29].

- Skoda Data set: This data set describes the activities of assembly-line workers in a car production environment. The data set considers the recognition of 11 activity classes performed for one of the quality assurance check-points of the production plan. In the study, one subject wore nineteen 3D accelerometers on both arms and performed a set of experiments using sensors placed on the two arms of a tester (10 sensors on the right arm, and 9 sensor on the left arm). The Skoda data set has been employed to evaluate deep learning techniques in sensor networks, which makes it an appropriate data set to evaluate our proposed substitute autoencoder framework.
- Hand Gesture Data set: In this data set, the sensory data from hand gestures are recorded for two subjects. The data is recorded using an accelerometer and gyroscope worn by the subjects. The data sets consist of 12 classes/activities performed. The 12 classes include 8 regular gestures and 3 gestures for playing tennis. The data set also includes a null activity where no gesture is performed.

B. Experimental Setting

The Skoda and Hand gesture data sets are human activity and context recognition data sets. The data sets have different classes. We divided the classes into three different subsets: desired, sensitive, and non-sensitive subsets. We mentioned the desired and non-sensitive classes to the server. The user did not provide the sensitive subset to the server but only to the trusted authority. To send the transformed data set to the server, the following steps were performed:

- First we took the sliding window size d = 30 and step size w = 3. Then, each of the classes/activities are mapped to numbers starting from zero. Afterwards, using the train split we created the training and testing data set. The training and testing data sets are in the shape of (samples, features, window size).
- In the second step, we created three data sets: White (desired), Black (sensitive), Gray (non sensitive) subsets. These subsets will be used to train the substitute autoencoder.
- We created a transformed data set, which will hide the sensitive attributes. To hide the sensitive attributes we substitute the non-sensitive or gray data set with the black data set. After substitution, we added gaussian noise to the transformed black subset using the following commands:

  rnd_ idx_ train = np.random.choice(g_ train_ data.shape[0], b_ train_ data.shape[0], replace=False)

  b_ train_ transformed = g_ train_ data[rnd_ idx_ train,:]

  b_ train_ transformed = x_ train_ transformed + 0.5 * *np.random.normal(loc=0.0, scale=1.0, size=b_ train _ transformed.shape)*

  b_ train_ transformed = np.clip(b_ train_ transformed, 0., 1.)

  rnd_ idx_ test = np.random.choice(g_ test_ data.shape[0], b_ test_ data.shape[0], replace=False)

  b_ test_ transformed = g_ test_ data[rnd_ idx_ test,:]

  b_ test_ transformed = x_ test_ transformed + 0.5 * *np.random.normal(loc=0.0, scale=1.0, size=b_ test _ transformed.shape)*

  b_ test_ transformed = np.clip(b_ test_ transformed, 0., 1.)

  Now, the transformed data set consists of white, grey and black transformed subsets. We substitute the black subset with the grey subset. Then, the substitute autoencoder maps the transformed data set to the original data set. The output will be the transformed data set where the black subset has been substituted by the grey subset.

VI. Results

To analyze the efficacy of our model we used different flavors of autoencoders. We analyzed our model with the deep autoencoder [30], convolutional autoencoder [31], and finally analyzed with the LSTM autoencoder [32]. Our data sets are time series data that contain repeated patterns, and thus, we chose deep ,convolutional and LSTM autoencoders. **Deep autoencoder:** We first reshaped the Skoda and the Hand Gesture data sets from 3d to 2d shape. Then, we used three layers for encoding as well as three layers for decoding. The activation function for the input and the output layers is 'Linear'. For all hidden layers we used the Scaled exponential Linear Unit (Selu). We used the Mean Square Error (MSE) as loss function.

Convolutional Denoising Autoencoder: Here we used a 1d convolutional autoencoder. We used 'relu' as the activation function in all layers except the output layer. In the output layer, we applied 'sigmoid' as the activation function. Here, we also used MSE as the loss function.

LSTM Denoising Autoencoder: LSTM Autoencoder can learn a compressed representation of sequence data and has been used on video, text, audio, and time series sequence data. Inspired by its application in time series data analysis, we implemented LSTM in our model. Here we reshaped the data into (samples, 1, windowsize$\times$features). Hence, for the left hand the data is reshaped into (samples, 1, 30×54), and for the right hand the data is reshaped into (samples, 1, 30×60) since the right hand and the left hand have 60 and 54, respectively.

For the Hand gesture data set for both Subject 1 and 2 the data is reshaped into (samples, 1, 30×15).

Finally, the output of each of the different autoencoders is sent to the server for classification. The server is a machine learning platform, which contains a convolutional neural network. However, we need to check the efficiency of both the original and the transformed data set. If the transformed data set does not have the efficiency as the raw data set, then it is referred to as a misleading data set. The data set will be of no use to the server. Thus, we evaluated the performance of both the original and the transformed data sets with the convolutional autoencoder in the server.

For the Skoda data set, the results are summarized in Table I, Table II and Table III. Table I shows the F1-score of both the original and the transformed data set for the CNN denoising autoencoder. Table II shows the F1-score of both the original and the transformed data set for the Deep denoising autoencoder. Table III summarizes the F1-score of the LSTM denoising autoencoder.

TABLE I
F1-SCORE FOR CNN AUTOENCODER OF SKODA DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

Hand	List of subsets	OF1	TF1
	Sw={4,8,9,10}	94.19	18.13
	Sb={1,5,6,7}	87.21	00.10
Left	Sg={0,2,3}	90.95	62.83
	Sw={0,2,3}	90.95	65.69
	Sb={1,5,6,7}	87.21	02.72
Left	Sg={4,8,9,10}	94.19	15.56
	Sw={4,8,9,10}	94.19	15.37
	Sb={1,5,6}	87.58	00.04
Left	Sg={0,2,3,7}	90.27	60.68
	Sw={4,8,9,10}	96.26	14.98
	Sb={1,5}	92.47	04.26
Left	Sg={0,2,3,6,7}	90.18	57.48
	Sw={4,8,9,10}	96.42	11.18
	Sb={1,5,6,7}	91.41	00.31
Right	Sg={0,2,3}	88.30	63.36
	Sw={1,4,10}	95.90	11.18
	Sb={2,3,8,9}	91.41	00.31
Right	Sg={0,5,6,7}	88.30	63.36
	Sw={1,4,10}	95.90	01.16
	Sb={2,3,9}	89.87	07.33
Right	Sg={0,5,6,7,8}	90.76	52.36
	Sw={4,9}	96.87	26.45
	Sb={1,2,3}	89.44	09.63
Right	Sg={0,5,6,7,8,10}	91.17	52.14

For the Hand Gesture data set, the results are summarized in Table IV, Table V and Table VI. Table IV shows the F1-score of both the original and transformed data set for the CNN denoising autoencoder. Table V shows the F1-score of both the original and the transformed data set for the Deep denoising autoencoder. Table VI summarizes the F1-score of the LSTM denoising autoencoder.

TABLE II
F1-SCORE FOR DEEP AUTOENCODER SKODA DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

Hand	List of subsets	OF1	TF1
	Sw={4,8,9,10}	95.47	56.54
	Sb={1,5,6,7}	87.04	00.09
Left	Sg={0,2,3}	87.83	66.21
	Sw={0,2,3}	87.83	81.78
	Sb={1,5,6,7}	87.04	00.14
Left	Sg={4,8,9,10}	95.47	89.51
	Sw={1,3,5,7}	93.50	76.37
	Sb={0,2}	88.90	12.17
Left	Sg={4,6,8,9,10}	94.76	83.27
	Sw={1,5,6,7}	87.04	13.46
	Sb={0,2,3}	87.83	52.50
Left	Sg={4,8,9,10}	95.47	30.18
	Sw={2,3,9}	89.87	01.18
	Sb={0,5,6,10}	90.19	14.28
Right	Sg={1,4,7,8}	93.12	09.92
	Sw={2,3,9,10}	97.93	06.55
	Sb={0,5,6}	91.53	63.77
Right	Sg={1,4,7,8}	97.90	05.48
	Sw={2,3,9}	89.87	82.17
	Sb={0,5,6}	89.53	01.14
Right	Sg={1,4,7,8,10}	94.77	90.10
	Sw={2,3,9,10}	97.93	19.60
	Sb={0,4,5,6}	92.56	02.28
Right	Sg={1,7,8}	98.12	19.40

TABLE III
F1-SCORE FOR LSTM AUTOENCODER SKODA DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

Hand	List of subsets	OF1	TF1
	Sw={4,8,9,10}	95.95	93.89
	Sb={1,5,6,7}	88.00	00.04
Left	Sg={0,2,3}	91.89	92.05
	Sw={0,2,3}	91.89	91.80
	Sb={1,5,6,7}	88.00	00.26
Left	Sg={4,8,9,10}	95.95	93.28
	Sw={4,8,9,10}	95.95	94.35
	Sb={1,5,6}	87.52	00.19
Left	Sg={0,2,3,7}	91.49	91.13
	Sw={4,8,9,10}	95.95	93.91
	Sb={1,5}	92.88	00.34
Left	Sg={0,2,3,6,7}	89.88	88.33
	Sw={4,8,9,10}	96.42	94.72
	Sb={1,5,6,7}	91.41	04.14
Right	Sg={0,2,3}	88.33	87.55
	Sw={1,4,10}	95.90	95.26
	Sb={2,3,8,9}	91.37	00.05
Right	Sg={0,5,6,7}	89.59	89.46
	Sw={1,4,10}	95.90	95.55
	Sb={2,3,9}	89.87	00.02
Right	Sg={0,5,6,7,8}	90.76	90.58
	Sw={2,3,9}	89.87	86.63
	Sb={1,4}	96.41	00.66
Right	Sg={0,5,6,7,8,10}	91.17	89.79

TABLE IV
F1-SCORE FOR CNN AUTOENCODER HAND GESTURE DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

subject	List of subsets	OF1	TF1

1	Sw={1,2,3}
	Sb={0,8,9,10,11}
	Sg={4,5,6,7}
		90.48
		92.30
		90.28
			33.51
			63.83
			0.34

1	Sw={0,4,5,6,7}
	Sb={1,2,3}
	Sg={8,9,10,11}
		92.63
		90.40
		91.14
			60.45
			31.98
			1.93

1	Sw={8,9,10,11}
	Sb={1,2,3}
	Sg={0,4,5,6,7}
		91.14
		90.40
		92.63
			0.17
			16.55
			58.87

1	Sw={4,8,9,10}
	Sb={1,5}
	Sg={0,2,3,6,7}
		91.14
		91.48
		92.29
			2.73
			45.01
			55.24

2	Sw={1,2,3}
	Sb={0,8,9,10,11}
	Sg={4,5,6,7}
		92.00
		92.55
		94.02
			10.83
			70.63
			2.88

TABLE V
F1-SCORE FOR DEEP AUTOENCODER HAND GESTURE DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

Subject	List of subsets	OF1	TF1

1	Sw={1,2,3}
	Sb={0,8,9,10,11}
	Sg={4,5,6,7}
		91.58
		91.32
		90.79
			83.67
			65.34
			84.70

1	Sw={1,8,9,10,11}
	Sb={2,3}
	Sg={0,4,5,6,7}
		89.37
		92.68
		91.58
			81.78
			01.33
			89.96

1	Sw={8,9,10,11}
	Sb={1,3}
	Sg={0,2,4,5,6,7}
		89.50
		91.58
		91.58
			88.01
			04.38
			90.58

1	Sw={8,9,10,11}
	Sb={1,3}
	Sg={0,2,4,5,6,7}
		89.50
		91.86
		91.58
			88.80
			0.5
			88.80

2	Sw={8,9,10,11}
	Sb={1,3}
	Sg={0,2,4,5,6,7}
		92.55
		92.02
		93.49
			89.82
			01.13
			91.33

2	Sw={1,3,10,11}
	Sb={8,9}
	Sg={0,2,4,5,6,7}
		92.18
		92.52
		93.49
			88.45
			06.76
			92.75

2	Sw={1,8,10,11}
	Sb={2,3}
	Sg={0,4,5,6,7,9}
		93.40
		91.46
		93.42
			88.56
			0.18
			92.95

2	Sw={1,3,8,10}
	Sb={2,4,11}
	Sg={0,5,6,7,9}
		93.6
		89.24
		93.73
			86.22
			0.74
			93.47

TABLE VI
F1-SCORE FOR LSTM AUTOENCODER HAND GESTURE DATA SET (OF1 STANDS FOR ORIGINAL DATA SET AND TF1 STANDS FOR TRANSFORMED DATA SET)

Subject	List of subsets	OF1	TF1

1	Sw={1,2,3}
	Sb={0,8,9,10,11}
	Sg={4,5,6,7}
		39.98
		52.52
		81.08
			38.78
			00.37
			80.41

1	Sw={0,4,5,6,7}
	Sb={1,2,3}
	Sg={8,9,10,11}
		91.64
		91.86
		90.83
			89.33
			00.33
			88.33

1	Sw={8,9,10,11}
	Sb={1,2,3}
	Sg={0,4,5,6,7}
		92.04
		92.06
		92.00
			85.99
			00.21
			88.77

1	Sw={8,9,10,11}
	Sb={1,3}
	Sg={0,2,4,5,6,7}
		90.20
		91.27
		91.63
			89.82
			01.20
			88.77

2	Sw={8,9,10,11}
	Sb={1,3}
	Sg={0,2, 4,5,6,7}
		92.55
		92.02
		93.49
			89.82
			01.13
			91.33

2	Sw={1,3,10,11}
	Sb={8,9}
	Sg={0,4,5,6,7,9}
		93.40
		91.46
		93.42
			88.56
			00.18
			92.95

2	Sw={1,3,8,10}
	Sb={2,4,11}
	Sg={0,5,6,7,9}
		93.60
		89.24
		93.73
			86.22
			00.74
			93.47

2	Sw={2,3,9}
	Sb={1,4}
	Sg={0,5,6,7,8,10}
		89.87
		96.41
		91.17
			86.63
			00.66
			89.79

Finally, we plot the confusion matrix of different substitute autoencoders of two data sets to evaluate the performance of our model. Figure 1, 2 and 3 shows the transformed Skoda data sets using CNN, multilayer, and LSTM as autoencoder, respectively. The confusion matrices are for the left hand Skoda data set with Sw={4, 8, 9, 10}, Sb={1, 5, 6, 7}, and Sg={0, 2, 3}. For the Skoda data set, we can say that the LSTM substitute autoencoder performs best. We can see that the white subsets of the LSTM data set has a low false positive rate. The white subset is the desired subset, which will be used by the third party. Thus, we can use the LSTM autoencoder to hide the sensitive subsets. The other two substitute autoencoders have a very high false positive rate as compared to LSTM. Also, by looking at the confusion matrix of the LSTM and the original data set we can say that the true positive rate of the white (desired) data set is almost equal.

Figure 4, 5 and 6 shows the transformed Hand gesture data sets of CNN, multilayer, and LSTM as autoencoder, respectively. The confusion matrices are for subject 1 data set with Sw={8, 9, 10, 11}, Sb={1, 2, 3}, and Sg={0, 2, 4, 5, 6, 7}. For the Hand Gesture data set of Subject 1 we can say that the LSTM substitute autoencoder performs best. We can see that the white subsets of the LSTM data set has a low false positive rate. The white subset is the desired subset, which will be used by the third party. Thus, we can use the LSTM autoencoder to hide the sensitive subsets. The other two

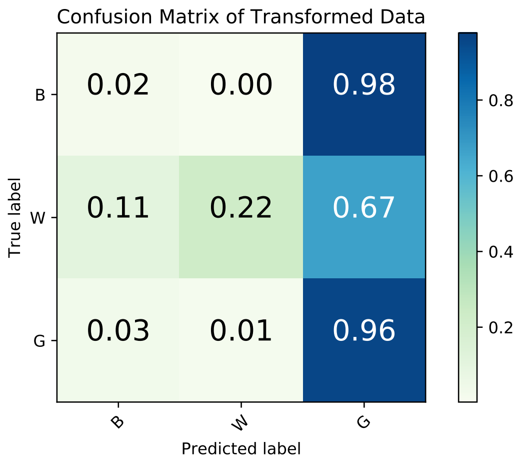

Fig. 1. Confusion Matrix of CNN autoencoder of Skoda Data Set

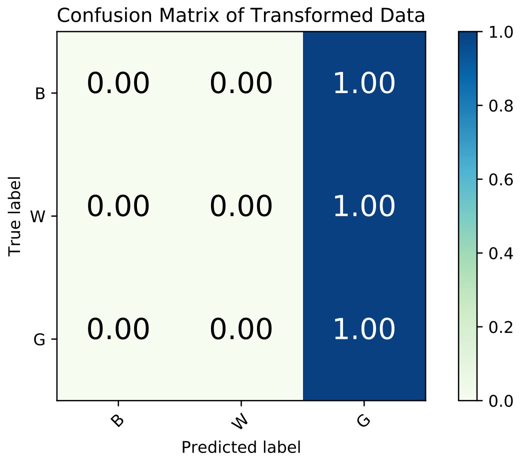

Fig. 2. Confusion Matrix of multilayer autoencoder of Skoda Data Set

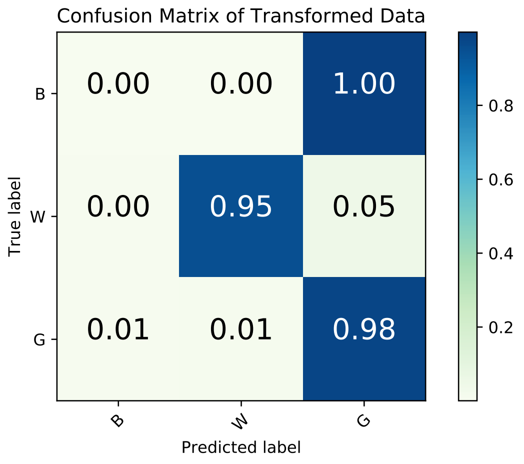

Fig. 3. Confusion Matrix of LSTM autoencoder of Skoda Dataste

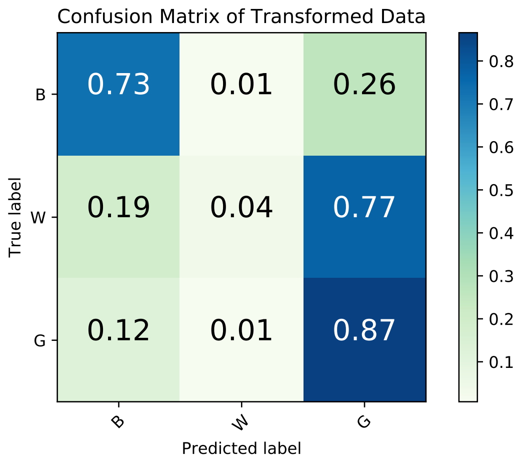

Fig. 4. Confusion Matrix of CNN autoencoder of Hand Gesture Data Set

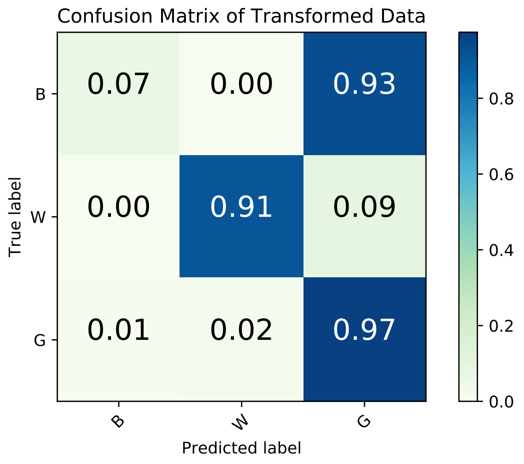

Fig. 5. Confusion Matrix of Multilayer autoencoder of Hand Gesture Data Set

substitute autoencoders have a very high false positive rate as compared to LSTM. Also, by looking at the confusion matrix of the LSTM and the original data set we can see that the true positive rate of the white (desired) data set is almost equal.

VII. Conclusion

Privacy preserving of time series data is a very challenging issue especially when part of the information is private. Here, in this paper we focused on how to hide the sensitive part of time series data. We divided the data into three parts. The parts are sensitive, non-sensitive and desired. The desired part will be used for further analysis by the server. The user as well as the server is not bothered about the non-sensitive portion. At the same time, the user does not want the server to know about the sensitive portion. The sensitive portion will not be used by the server for further analysis. Thus, we substitute the sensitive portion with the non-sensitive portion. We used the property of denoising autoencoder. However, instead of only using external noise to hide data, we substitute the sensitive

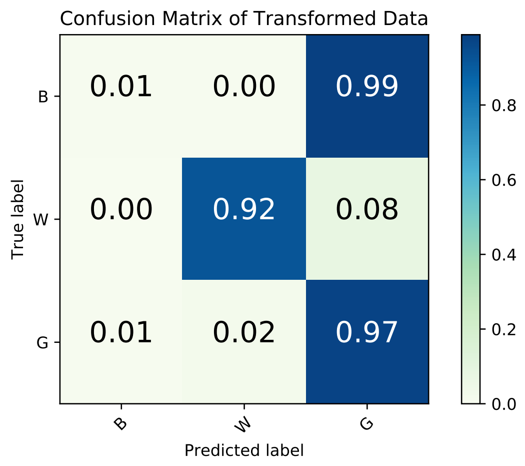

Fig. 6. Confusion Matrix of LSTM autoencoder of Hand Gesture Data Set

information with non-sensitive information. After that we add noise to the transformed sensitive information. We named the autoencoder as substitute autoencoder.

To measure the efficacy of our model, we used three different autoencoders: Deep autoencoder, LSTM autoencoder and Convolutional autoencoder for our two different data sets. We measured the F1-score for each of the three autoencoders with both the original and the transformed data sets. The F1-score was also analyzed with the three different subsets. We found Skoda data set of the left hand with subsets Sw = {4, 8, 9, 10}, Sb = {1, 5, 6}, Sg = {0, 2, 3} achieved the best F1-score. By comparing the original F1-score with the transformed F1-score, the black subset has been reduced from 88% to almost 0% (0.04%), while at the same time the gray and the white subset is almost the same as the original subset. For the right hand data set with subsets Sw = {1, 4, 10}, Sb = {2, 3, 8, 9}, Sg = {0, 5, 6, 7} resulted in a good F1-score. The F1-score for the black subset has been reduced from 91.37% to almost 0% (0.04%). For the Hand gesture data set of Subject 1 with subsets Sw = {1, 2, 3}, Sb = {0, 8, 9, 10, 11}, Sg = {4, 5, 6, 7} achieved the best F1-score. By comparing the original F1-score with the transformed F1-score, the black subset has been reduced from 53% to 0.37, while at the same time the gray and the white subset is almost the same as the original subset. Subject 2 with subsets Sw = {1, 3, 10, 11}, Sb = {8, 9}, Sg = {0, 4, 5, 6, 7, 9} resulted in the best F1-score. The F1-score for the black subset has been reduced from 91.46% to 0.18%. We also plotted the confusion matrix of the three autoencoders of the left hand Skoda data set and Subject 1 of the hand gesture data set. Overall, we see that the LSTM performed better than the deep autoencoder, and the convolutional autoencoder.

References

[1] M. Baccouche, et al., Sequential deep learning for human action recognition. International workshop on human behavior understanding. Springer, Berlin, Heidelberg, 2011.
[2] J. Dai, et al., Towards privacy-preserving recognition of human activities. 2015 IEEE international conference on image processing (ICIP). IEEE, 2015.
[3] CNN-wyzelab, http://www.cnn.com/2019/12/30/tech/wyze-data-breach/index.html, last accessed 12/30/2019.
[4] M. Joye, et al., A scalable scheme for privacy-preserving aggregation of time-series data. International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, 2013.
[5] C. Dwork, et al., The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science 9.3-4: 211-407, 2014.
[6] J. Tang, et al., Privacy loss in apple's implementation of differential privacy on macos 10.12. arXiv preprint arXiv:1709.02753, 2017.
[7] C. Dwork, Differential privacy. Encyclopedia of Cryptography and Security: 338-340, 2011.
[8] M. Abadi, et al., Deep learning with differential privacy. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.
[9] MNIST, http://yann.lecun.com/exdb/mnist/, last accessed 12/30/2019.
[10] CIFAR, http://www.cs.toronto.edu/~kriz/cifar.html, last accessed 12/30/2019.
[11] J. H. Friedman, Stochastic gradient boosting. Computational statistics & data analysis 38.4: 367-378, 2002.
[12] F. Casino, et al., A k-anonymous approach to privacy preserving collaborative filtering. Journal of Computer and System Sciences 81.6: 1000-1011, 2015.
[13] S. Xiaoyuan, et al., A survey of collaborative filtering techniques. Advances in artificial intelligence 2009 .
[14] B. Hu, et al., Data sparsity: a key disadvantage of user-based collaborative filtering? Asia-Pacific Web Conference. Springer, Berlin, Heidelberg, 2012.
[15] S. Dave, Procedures for geomasking to protect patient confidentiality. ESRI international health GIS conference, 2004.
[16] K. H. Hamton, et al., Mapping health data: improved privacy protection with donut method geomasking. American journal of epidemiology 172.9: 1062-1069, 2010.
[17] S. Zhang, et al., The location swapping method for geomasking. Cartography and Geographic Information Science 44.1: 22-34, 2017.
[18] A. Evfimievski, Randomization in privacy preserving data mining. ACM Sigkdd Explorations Newsletter 4.2: 43-48, 2002.
[19] S. R. M. Oliveira, et al., Privacy preserving clustering by data transformation. Journal of Information and Data Management 1.1: 37-37, 2010.
[20] S. Pattanayak, et al., A Secure Access Authentication Scheme for Multi-server Environments using Neural Cryptography. Journal Of Information Assurance and Security 13.1: 56-65, 2018.
[21] S. Pattanayak, et al., Improving Data Privacy Using Fuzzy Logic and Autoencoder Neural Network. 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2019.
[22] V. V. Kumari, et al., Fuzzy based approach for privacy preserving publication of data. International Journal of Computer Science and Network Security 8.1: 115-121, 2008.
[23] M. Malekzadeh, et al., Replacement autoencoder: A privacy-preserving algorithm for sensory data analysis. arXiv preprint arXiv:1710.06564 2017.
[24] G. F. Hinton, et al., Reducing the dimensionality of data with neural networks. science 313.5786: 504-507, 2006.
[25] V. Pasal, et al., Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th international conference on Machine learning. ACM, 2008.
[26] S. Lawrence, et al., Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks 8.1: 98-113, 1997.
[27] Y. Kim, Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 2014.
[28] Skoda-data set, har-dataset.org, last accessed 12/29/2019.
[29] A. Bulling, et al., A tutorial on human activity recognition using body-worn inertial sensors. ACM Computing Surveys, vol. 46, no. 3, pp. 33:133:33, 2014.
[30] R. M. Alguliyev, et al., Privacy-preserving deep learning algorithm for big personal data analysis. Journal of Industrial Information Integration 15: 1-14, 2019.
[31] M. Chen, et al., Deep features learning for medical image analysis with convolutional autoencoder neural network. IEEE Transactions on Big Data, 2017.
[32] A. Gensler, et al., Deep Learning for solar power forecasting-An approach using AutoEncoder and LSTM Neural Networks. 2016 IEEE international conference on systems, man, and cybernetics (SMC). IEEE, 2016.