Deep Learning from Distributed Images on a Privacy Preserving Internet of Autoencoders

Hojjat Salehinejad*†, Shahrokh Valaee*, Tim Dowdell†, and Joseph Barfett†
*Department of Electrical & Computer Engineering, University of Toronto, Canada
†Medical Imaging Department, St. Michael's Hospital, University of Toronto, Canada
salehinejadh@smh.ca, valaee@ece.utoronto.ca, dowdellt@smh.ca, barfettj@smh.ca

***Abstract*—Very large datasets are sitting in house all over the globe. Hospitals for example host a large number of labeled medical images, medical notes, as well as laboratory data. Patients often maintain their personal medical data, such as drug prescriptions or test results, on home computers and portable devices. This valuable data can be used for developing deep learning models, which are very data intensive, to automate diagnostic procedure or pathology detection models for many types of disease. Privacy issues are the most important bottleneck to sharing such data. In this paper, we propose an internet of autoencoders to extract and transmit only the latent space compressed representation features of data, such as medical images. This method keeps the original image in house, while transmitting only the key features of the images necessary for training a deep learning model to a data repository. In this paper, we report preliminary performance results of our experiments on MNIST and a set of multimodal medical images to detect the data class of each image modality. The results show high performance of the autoencoders to extract latent space compressed representations of images. The classifier microservice at the server, trained with extracted feature vectors, has very high classification accuracy for both MNIST and medical images datasets.**

***Index Terms*—Autoencoder, deep learning, distributed learning, internet of things, medical images, privacy.**

I. Introduction

Personally identifiable information (PII) refers to any data or information that can potentially identify a specific individual, or in a medical environment a patient. Any kind of information that could be used to distinguish one person from another and can be used for deanonymizing of anonymous data falls under PII category. PII data itself can be categoreized into sensitive and non-sensitive information. The first refers to information which could result in harm to the individual or organization who holds/owns the data, such as hospital regarding medical images or disease states. This type of PII needs to be highly secured and encrypted in transit and storage [1]. The later refers to data that can be transmitted or stored in an unencrypted form without resulting in harm to an individual or organization. Some examples are public records, phone books, and corporate directories.

The internet of things provides an infrastructure for data sharing between multiple devices, individuals, and organization. However, due to security and privacy issues, many individuals and organizations, such as hospitals, cannot share their resources for public or research benefit. In this case, very useful information that can be used for developing new technologies, such as automated pathology methods for early cancer detection, will remain in house where it cannot be accessed.

With the current interest in machine learning and big data, many medical resources are limited to the house they belong to and are able to contribute to a larger learning system. Machine learning models, particularly deep artificial neural networks [2], require access to very large datasets to improve their performance. Such models require training with very large and diverse datasets to increase their generalization, such that the model can perform well for unseen input data.

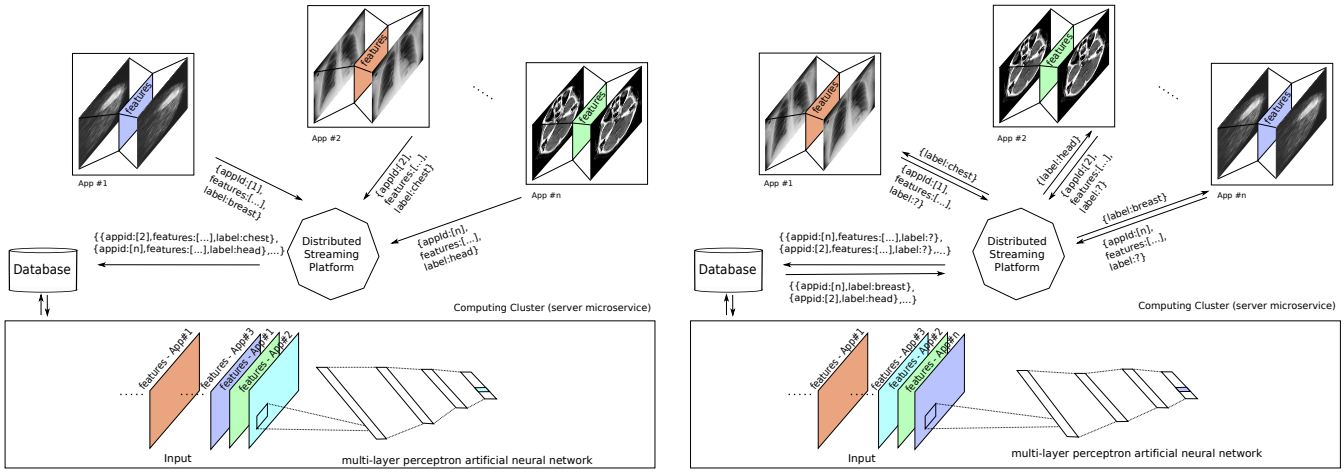
In this paper, we propose a new way of sharing data on an internet of things infrastructure using an internet of autoencoders. In this architecture, an autoencoder microservice runs at each user's device. The autoencoder can extract important features that exist in the image, i.e., latent space compressed representation features, and transmit them to a central computing cluster which can provide a service back to the user, such as detection of abnormalities in the captured image by the remote device at home. This method assures that only an abstract version of the data is leaving the local storage, which has no meaning to human eye on its own. This architecture also enables radiologists at hospitals all around the globe performing interpretation on medical images (i.e., labeling the images) to send the latent space compressed representation features of their medical images along with the corresponding data classes (i.e., label) to a computing cluster for training automated pathology machine learning models, to name one example.

Next section provides a brief overview on the literature in learning from distributed data and related privacy matters. The proposed method is discussed in Section III and the experimental results are presented in Section IV. The paper is concluded in Section V.

II. Background

Different frameworks are proposed in the literature to address the privacy preserving of machine learning models for data classification such as [3]–[5], [8]. The proposed method in [5] keeps the training samples private while enabling accurate construction of logistic regression models. A scenario

The authors thank the support of NVIDIA Corporation with the donation of the Titan X GPUs used for this project.



(a) Model architecture in training mode.

(b) Model architecture in service mode.

Fig. 1: Proposed model architecture in training and service modes. The messages are just for illustration purpose. The structure of messages is discussed in Sections III-D and III-E. A client microservice is denoted with App.

is studied in [6], where multiple data holders try to find predictive models from their joint data without sharing their data with each other. In this method, the centralized learning task distributed among the data holder as local learning tasks, such that local learning is performed on locally available data. In another attempt for privacy-preserving in machine learning, the training data can be sliced and distributed based on an Apache Hadoop architecture [7]. This method uses some cryptographic operations at the reduce level of map-reduce procedure to achieve privacy-preservation [7]. In another approach, a privacy-preserving deep learning system is built to be trained over a combined dataset from different users, without revealing the original local data to a central server [9].

Some deep learning models in distributed mode are recently presented [10]. These models are focused on distributing the training load of deep learning models on multiple devices. In [11], a model is proposed, which is consisted of multiple parties to jointly learn a deep learning model for a given objective without sharing their input datasets. This model lets users train their model independently on their own datasets and selectively share small subsets of the key parameters of their model during training. A software framework called DistBelief is presented in [10], which utilizes computing clusters with thousands of machines to train large models. Neural network computations are performed on a set of encrypted data in [12]. This method uses encryption to communicate data between user and a computing cluster. An other approach is a distributed multinode synchronous stochastic gradient descent algorithm, without altering hyperparameters, or compressing data, or altering algorithmic behavior is proposed in [13].

The proposed method in this paper is not a distributed deep learning framework. However, it uses distributed data, which their features are extracted using an autoencoder, to train a centeralized deep learning model. An autoencoder is an unsupervised machine learning model that uses a learning method such as backpropagation to learn the main features of a given input, while using the same input as output of the model. Autoencoders are successfully used for various application such as HIV classification [14], speech recognition [15], reinforcement learning [16], medical image denoising [17], and mammogram compression [18].

III. Proposed Model Architecture

The proposed architecture is consisted of three main layers, autoencoder microservice at client, distributed streaming platform, and a classifier microservice at server. These layers of microservices work together in two modes, which are learning mode and service mode. Each microservice layer and the modes are described as follow.

A. Autoencoder Microservice at Client Side

Each client in the proposed architecture in Figure 1a, can be a radiologist sitting at a hospital or a patient who is using a personal device. Privacy is highly important for this data and clients require high level of protection of their data to ensure privacy. Meanwhile, their images need to be analyzed for diagnostic purposes, as an example. To ensure this objectives, the image can be passed to an autoencoder microservice, which is implemented as an application on their smart device or computer. The proposed solution is to utilize an autoencoder to extract features of the image and only transmit these features to the classifier microservice at the server side in the learning mode.

As it is demonstrated in Figure 2, the convolution layer in a convolutional autoencoder can extract features for the feature map m from the input image $I^{(s)}_{U\times V}$ from user s using a convolution kernel of size $K \times K$ as

$$h^{(m)}_{u,v} = \sigma\Big(\sum_{i=0}^{K-1}\sum_{j=0}^{K-1} I_{i,j}\cdot w^{(m)}_{i,j} + b^{(m)}_{u,v}\Big), \qquad (1)$$

for $u \in \{0, U-1\}$ and $v \in \{0, V-1\}$, where $\sigma(\cdot)$ is an activation function such as rectified linear unit (ReLU)

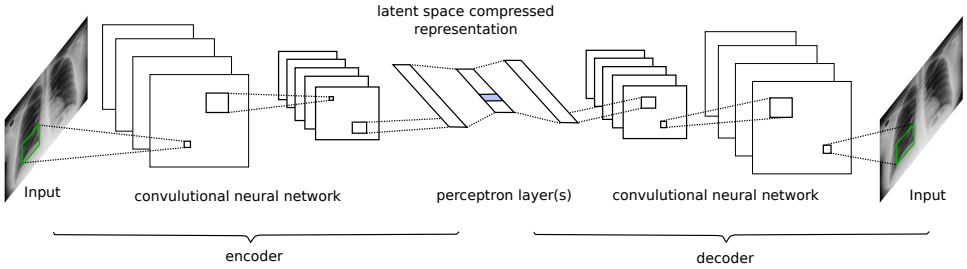


Fig. 2: Convolutional autoencoder microservice at client side.

and $\mathbf{w}^{(m)}$ is the weight matrix of feature map m. A max-pooling layer after the convolution layer down-samples the latent representation by a constant factor, usually taking the maximum value over non overlapping sub-regions such as

$$O_{i,j} = max\{h^{(m)}_{q,r}\}, \tag{2}$$

for $q, r \in \{(Li, Lj), ..., (L(i+1)-1, L(j+1)-1)\}$, where the max-pooling kernel is of size $L \times L$. This operation helps to obtain translation-invariant representations. Then, the max-pooled features are reshaped as a vector $\mathbf{f}$ and are fed to a perceptron layer for dimension reduction such as

$$y_i = \sum_{j=0}^{|\mathbf{f}|}(f_j \cdot w_{j,i} + b_i) \tag{3}$$

where $|\mathbf{f}|$ is the length of $\mathbf{f}$ and $\mathbf{w}$ is the weight of connections between the layers $\mathbf{f}$ and $\mathbf{y}$.

The next phase of the autoencoder it decoding the reduced representation vector $\mathbf{y}$ to reconstruct the original image $I^{(s)}$. This stage is conducted by reversing the steps discussed in the encoder phase. The objective is to minimize the loss between the original image and the reconstructed image $\tilde{I}^{(s)}$ such as

$$\mathcal{L} = ||I^{(s)} - \tilde{I}^{(s)}||^2. \tag{4}$$

A convolutional autoencoder can have multiple convolution layers and hidden perceptron layers. The latent space compressed representation of the image $I^{(s)}$ is $\mathbf{y}$, which is sent to the classifier microservice at server through a distributed streaming platform.

B. Distributed Streaming Platform

The distributed streaming platform in Figure 1a builds the pipelines for data transfer between clients (message producers) and the server (message consumer). This real-time streaming platform manages the queue of the produced messages and guarantees that a copy of the produced messages (i.e., latent space compressed representation of an image) by clients will be stored for classification at server side. It also has a pipeline to a permanent database.

C. Classifier Microservice at Server Side

The classifier at server side in Figure 1a, in learning mode, reads the features and corresponding labels in batches from the database for training. Since the input data is already feature representations of original images, the classifier can be a multi-layer perceptron (MLP) or a more complex architecture, depending on the complexity of the dataset and learning task. For example, in Figure 1a an MLP neural network with two hidden layers are used. In this approach, for an input feature map $\mathbf{x}$ we have

$$h^1_i = \sum_{j=0}^{|\mathbf{x}|}(x_j \cdot w^{h1}_{j,i} + b^{h1}_i), \tag{5}$$

and

$$h^2_i = \sum_{j=0}^{|\mathbf{h}^1|}(h^1_j \cdot w^{h^2}_{j,i} + b^{h^2}_i), \tag{6}$$

where h^1_i and h^2_i are the outputs of the first and second hidden layers for unit i, respectively. The output of the classifier at output layer is the probability distribution over P data classes (labels) given by a softmax function $\phi(\cdot)$ such as

$$y_p = \phi(\sum_{l=1}^{|\mathbf{h}^2|}(h^2_l \cdot w^o_{l,p} + b^o_p) \tag{7}$$

where $|h^2|$ is the length of second hidden layer, $w^o_{l,p}$ is the weight of connection from feature l at the hidden layer h^2 to label p in the output layer o. The softmax activation function defined as

$$\phi(z_p) = \frac{e^{z_p}}{\sum_{j=1}^{P} e^{z_j}} \ \forall \ p = \{1, ..., P\}. \tag{8}$$

The classifier can work in offline and online modes. In offline mode, the classifier just receives inputs and predict the corresponding label(s). In online mode, the classifier learns from the stored features in the database and updates the classifier parameters in offline mode on a regular time interval. Therefore, these two services can run in parallel on two independent computing clusters, with a shared memory.

D. Learning Mode

Clients can send messages in learning or service modes. In learning mode, as demonstrated in Figure 1a, they send the extracted latent space compressed representation of the image as well as the corresponding class (label). This can often happens when radiologists, who are working in different locations and cannot release the original image due to privacy issues, contribute to the learning by sending their interpretation

from the image (i.e., the label) along with the features. This type of messages are treated as trusted messages at the server side and are used for training the model.

The messages are in JavaScript Object Notation (JSON) data interchange format. A message in learning mode from client to server has the following JSON structure

```
{"message": {
  "clientID": "app2",
  "label": "chest x-ray"
  "value": [0.4,0.9,...,0.3],
  "mode": "learning",
  "timestamp": "2017-04-28 13:10:02"
}}
```

and the server upon receipt of the message acknowledges it.

E. Service Mode

The proposed architecture in service mode is presented in Figure 1b. A message in service mode has the following JSON structure

```
{"message": {
  "clientID": "app1",
  "label": ""
  "value": [0.2,0.7,...,0.4],
  "mode": "service",
  "timestamp": "2017-04-29 12:16:22"
}}
```

where the "mode": "service" indicates the client is waiting for the label(s) of the sent features. In this case, the features are passed to the offline classifier microservice for classification and the results is sent back to the client microservice as

```
{"message": {
  "clientID": "app1",
  "label": "chest x-ray"
  "timestamp": "2017-04-29 12:20:42"
}}
```

where the image is detected as "chest x-ray".

Since we are using a real-time distributed streaming platform and the classifier can work in offline and online modes, the server can serve both clients in learning mode and service mode.

IV. Experiments

In this section, we evaluate performance of the autoencoders to extract latent space compressed representation of images from various classes and at the classifier microservice in server side, we evaluate performance of detecting the correct data class from given feature vectors.

A. Data

For the experiments we use the MNIST dataset [19] and a set of medical images. The MNIST is a database of handwritten digits for digits 0 to 9. The medical images dataset is consisted of various modalities, which includes abdominal magnetic resonance imaging (MRI), breast, chest, head, lung, mammogram, and pelvis.

B. Setting

The autoencoder utilized in each client microservice for experiments is consisted of one convolutional layer followed with a max-pooling layer and an MLP with one hidden layer at the encoder side. The decoder side is the reverse of architecture in the encoder. The convolution kernel size is set to 5×5 and the max-pooling kernel is set to 2×2.

We utilize the Kafka distributed streaming platform to build real-time data pipelines and streaming apps. This framework is horizontally scalable and fault-tolerant.

The classifier microservice at server side is an MLP with two hidden layers. It reads the received latent space compressed representations in mini-batches of 64. The feature vectors are of length 784. The first to third hidden layers are vectors of length 256, 128, and 64, respectively. The output layer has P units which represent the number of data classes (labels) per dataset. Both the autoencoders and classifier use Adam optimizer and utilize adaptive learning rate. All the microservices are virtualized on a machine with two NVIDIA TITAN X graphical processing units (GPUs).

The experiments are conducted on MNIST and medical datasets independently. The network hyper-parameters are set based on a grid search. In the experiments, five client microservices are connected to the Kafka distributed platform. To train the classifier, each client sends feature vectors of 1,000 images with corresponding classes. Once the classifier is trained, the clients start randomly sending feature vectors in service mode from a test dataset. The test dataset has never been visited by the classifier. Total number of samples in the test dataset is 1,000.

C. Results

In order to visualize latent space compressed representation features and the corresponding reconstructed images, a randomly selected image from each data class of MNIST and medical multimodal images datasets are selected and fed to the convolutional autoencoder, illustrated in Figure 2. The reconstructed images from MNIST dataset shown in the third row of Figure 3 show reasonable reconstruction precision of the original images, such that the edges and high level presentation of the image is well reconstructed. The reconstructed images have some degree of noise (i.e., difference) comparing with the original images. This difference is not highly important, since the objective is the extract the principal features of the image and the reconstruction accuracy can be improved by better training of the autoencoder.

The middle row of Figure 3 show latent space compressed representations of the images. These images are visually highly

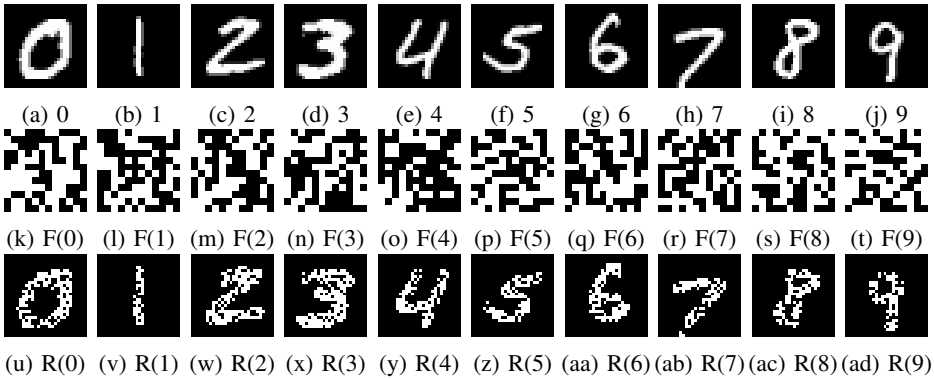


Fig. 3: Samples from the MNIST dataset, corresponding features (F(·)), and reconstructed output (R(·)) using a multi-layer perceptron autoencoder.

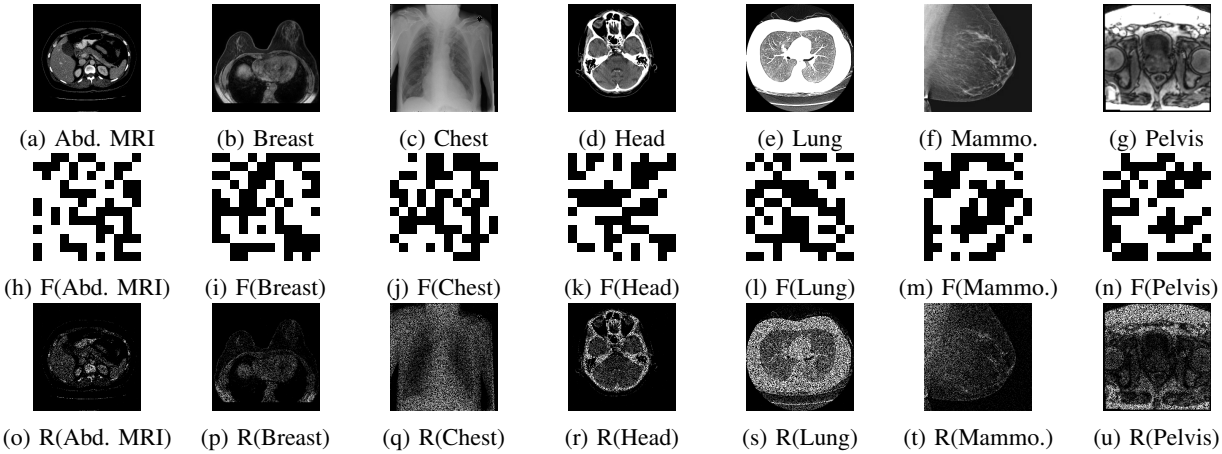


Fig. 4: Samples from the medical images dataset, corresponding features (F(·)), and reconstructed output (R(·)) using a multi-layer perceptron autoencoder.

TABLE I: Performance of the convolutional autoencoder in reconstruction of original images from latent space compressed representation features on MNIST and medical datasets.

Dataset	Data Class										Total
MNIST	0	1	2	3	4	5	6	7	8	9	98.68%
	97.47%	99.64%	98.59%	97.61%	98.79%	98.89%	99.15%	98.83%	98.79%	99.12%	
Medical	Abdominal MRI		Breast MRI		Chest	Head	Lung	Mammogram		Pelvis	90.64%
	91.63%		89.72%		87.39%	92.70%	91.29%	90.63%		91.12%	

TABLE II: Performance of the multi-layer perceptron artificial neural network in classification of extracted latent space compressed representation features on MNIST and medical datasets.

Dataset	Data Class										Total
MNIST	0	1	2	3	4	5	6	7	8	9	98.79%
	98.42%	99.10%	97.98%	98.62%	98.14%	99.13%	98.95%	98.91%	99.32%	99.39%	
Medical	Abdominal MRI		Breast MRI		Chest	Head	Lung	Mammogram		Pelvis	93.35%
	93.65%		91.19%		89.99%	94.80%	93.69%	94.96%		95.22%	

different from the original images and are not recognizable by human eye. Therefore, transmission of these feature should not be of any privacy or security concern. This case is extremely important for medical images, which the corresponding features are presented in second row of the Figure 4.

Performance of the autoencoder in extracting latent space compressed representation of each image is computed as

$$\delta = \frac{1}{U \times V}\left(\sum_{i=0}^{U-1}\sum_{j=0}^{V-1}(I_{i,j} - \tilde{I}_{i,j})^2\right)^{\frac{1}{2}}, \tag{9}$$

which is the Euclidean distance between the original I and reconstructed image $\tilde{I}$. The performance results for the MNIST and medical images are presented in Table I. The results show high performance of the autoencoder in learning the latent space compressed representation features and reconstructing the input image from these features. The medical images contain more details and are more challenging for the autoencoder to learn their features. However, the results show competitive performance of learning medical images comparing with the

simpler MNIST dataset.

Classification performance of the classifier is evaluated based on the correct prediction of the data class p as

$$\xi = \frac{1}{N}\sum_{i=1}^{N}|y_i(p) - t_i(p)|, \tag{10}$$

where N is size of the test dataset. The correct data class is stored in the one-hot encoded target vector $\mathbf{t} = (0, .., 0, \underset{p}{1}, 0, ..0)$ for class p. Table II shows performance of the classifier in classifying the latent space compressed representation features sent by the microservices at client side. The results show high performance of the MLP network with two hidden layers in predicting the corresponding data class of each latent space compressed representation features vector.

V. Conclusion

Training deep neural networks requires access to large datasets. Internet of things provides the environment for sharing data for that aim. However, many users such as patients or hospital are not comfortable with sharing their data, due to privacy and security reasons.

In this paper, we propose an architecture based on an internet of autoencoders to facilitate sharing of data without privacy concerns. The autoencoders can extract latent space compressed representation features of the images at user (client) side and transmit them to a computing clusters, which can train based on the features and provide classification and prediction services for users, such as for diagnostic purposes in medical imaging. Since the original images are not shared in this architecture, the privacy of data is well considered, while the users can contribute in learning of the system and take advantage of using the services. The preliminary results presented in this paper show promising performance of this system for further development as a secure way of sharing data on an internet of things infrastructure for various application.

References

[1] B. Krishnamurthy and C. E. Wills, "On the leakage of personally identifiable information via online social networks," in *Proceedings of the 2nd ACM workshop on Online social networks*. ACM, 2009, pp. 7–12.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] Q. Jia, L. Guo, Z. Jin, and Y. Fang, "Privacy-preserving data classification and similarity evaluation for distributed systems," in *Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on*. IEEE, 2016, pp. 690–699.

[4] P. Jain, M. Gyanchandani, and N. Khare, "Privacy and security concerns in healthcare big data: An innovative prescriptive." *Journal of Information Assurance & Security*, vol. 12, no. 1, 2017.

[5] Y. Gong, Y. Fang, and Y. Guo, "Private data analytics on biomedical sensing data via distributed computation," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 13, no. 3, pp. 431–444, 2016.

[6] K. Xu, H. Ding, L. Guo, and Y. Fang, "A secure collaborative machine learning framework based on data locality," in *Global Communications Conference (GLOBECOM), 2015 IEEE*. IEEE, 2015, pp. 1–5.

[7] K. Xu, H. Yue, L. Guo, Y. Guo, and Y. Fang, "Privacy-preserving machine learning algorithms for big data systems," in *Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on*. IEEE, 2015, pp. 318–327.

[8] Y. Aono, T. Hayashi, L. Wang, S. Moriai *et al.*, "Privacy-preserving deep learning: Revisited and enhanced," in *International Conference on Applications and Techniques in Information Security*. Springer, 2017, pp. 100–110.

[9] Y. A. Le Trieu Phong, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption."

[10] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in neural information processing systems*, 2012, pp. 1223–1231.

[11] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[12] R. Gilad-Bachrach, T. W. Finley, M. Bilenko, and P. Xie, "Neural networks for encrypted data," Nov. 7 2014, uS Patent App. 14/536,145.

[13] D. Das, S. Avancha, D. Mudigere, K. Vaidynathan, S. Sridharan, D. Kalamkar, B. Kaul, and P. Dubey, "Distributed deep learning using synchronous stochastic gradient descent," *arXiv preprint arXiv:1602.06709*, 2016.

[14] B. L. Betechuoh, T. Marwala, and T. Tettey, "Autoencoder networks for hiv classification," *Current Science*, pp. 1467–1473, 2006.

[15] T. Ishii, H. Komiyama, T. Shinozaki, Y. Horiuchi, and S. Kuroiwa, "Reverberant speech recognition based on denoising autoencoder." in *Interspeech*, 2013, pp. 3512–3516.

[16] S. Lange and M. Riedmiller, "Deep auto-encoder neural networks in reinforcement learning," in *Neural Networks (IJCNN), The 2010 International Joint Conference on*. IEEE, 2010, pp. 1–8.

[17] L. Gondara, "Medical image denoising using convolutional denoising autoencoders," in *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*. IEEE, 2016, pp. 241–246.

[18] C. C. Tan and C. Eswaran, "Using autoencoders for mammogram compression," *Journal of medical systems*, vol. 35, no. 1, pp. 49–58, 2011.

[19] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.