Anomaly Detection Using Federated Learning

Shubham Singh, Shantanu Bhardwaj, Hemlatha Pandey, and Gunjan Beniwal

Abstract Federated learning is the new tide that is being associated with machine learning territory. It is an attempt to enable smart edge devices to confederate a mutual prediction model while the training data is residing at the respective edge device. This facilitates our data to be more secure, use less bandwidth, lower latency, and power consumption. We exercise the concept of federated learning in our neural network autoencoder model to detect the anomaly. Anomaly detection is finding the unusual pattern in a given data stream which may be a false or mal-entry in the pool of transactions. It helps us to prevent many online theft and scams which are detected using state-of-the-art machine learning and deep learning algorithms. All this has to be implemented in smart edge devices that have enough computing power to train the models provided to them.

Keywords Federated learning · Machine learning · Deep learning · Anomaly detection · Autoencoders · PySyft

1 Introduction

2.5 quintillion bytes of data are produced every day in this modern world and this quantity will increase with each passing year. The data produced every day records our daily activities, our interests, our hobbies, our passwords, and all other important

S. Singh (✉) · S. Bhardwaj · H. Pandey · G. Beniwal
Maharaja Surajmal Institute of Technology, C-4 Janakpuri, Delhi 110058, India
e-mail: chauhanshubham089@gmail.com

S. Bhardwaj
e-mail: shantanub30@gmail.com

H. Pandey
e-mail: pandey.hema98@gmail.com

G. Beniwal
e-mail: gunjanbeniwal@msit.in

P. Bansal et al. (eds.), *Proceedings of International Conference on Artificial Intelligence and Applications*, Advances in Intelligent Systems and Computing 1164,
https://doi.org/10.1007/978-981-15-4992-2_14

details looked at by hackers and cybercriminals. So much data is sent over the network which becomes subject to cyberattacks. It also results in unnecessary bandwidth utilization.

So, to attain privacy and limit the bandwidth utilization, we use the concept of federated learning. Federated learning also known as private learning involves sharing of the trained model weights among the smart edge devices with data not being sent on a central server [1]. In traditional machine learning and deep learning approaches, we have to send data to a centralized server where a single model was trained.

In this paper, we are implementing anomaly detection [2] using federated learning. Anomalies are also termed as outliers, deviants, or anything different from the usual entry in a system that generates doubt of its authenticity. It can be encountered due to malfunctioning systems, cybercriminals, or just an entry that was never encountered previously in the system with various applications like credit card fraud detection, online payment, health monitoring, bug, or fault detection. In this paper, we have used autoencoders which are special types of neural networks used in unsupervised learning, which reconstruct input back to its output. We use this reconstruction loss as the basis to classify anomalies.

2 Related Work

Anomaly detection is a well-practiced application of machine learning and deep learning. There are many publications implementing different techniques to achieve the same goal of anomaly detection.

Such as work done by Martinelli et al. on electric power system anomaly detection [3] is based on neural networks. In their publication, they have mentioned that they have trained their autoencoder for 72 h and the dataset had 432 patterns. As a measure of comparison, they have taken the root mean square difference of their input vector and their output vector. They were able to achieve a praise-worthy error of 0.015 with 6000 epochs.

Ullah et al. work on real-time anomaly detection in dense crowded scenes [4] is also a great explanation of work done in the field of anomaly detection. They have pursued this topic to predict the panic situation in a crowded environment. A normal activity in the crowd is considered as walking and an abnormal activity in the crowd is considered running. The dataset used is available online for research purposes. In their publication, they have mentioned the use of MLP neural network for detecting the irregular pattern in a crowded video. They have used the motion features of the crowd which have been derived from the velocity magnitude of the corner feature as it is hard to consider velocity of every pixel of an individual in a crowd. Motion properties which are derived from corner features are taken as input to MLP. The cost function gives the squared error between the desired and calculated output vectors.

Dau et al. gave a detailed analysis on anomaly detection using replicator neural networks trained on examples of one class [5]. They showed how by using only one class, we can detect anomalies in various datasets. By using six different datasets

for their experiment, they produced impressive results in comparison to the previous work done on the same datasets as mentioned in their publications. The applications of anomaly detection can be seen in various fields such as credit card fraud detection, health monitoring, network intrusion, video surveillance, and other cybercrimes.

There are several autoencoders in the industry that can detect different anomalies in the data. There has been a thorough explanation of autoencoders in [6].

2.1 Federated Learning

The conventional approach of training a neural network requires a sole replica of the model and the whole training dataset is in one place. Generally, the data that is recorded by the edge devices need to be sent to the global-shared server for training purposes and the by-product network weights need to be sent back to the devices but these devices usually possess finite bandwidth and periodical connections to the globally shared server. However, in federated learning, we can train the model on each device separately and is resemblant to the data parallelism model.

The federated learning technique allows us to take advantage of the shared training model without the demand for storing it at a central server. The data is distributed over numerous clients that are connected to a mutual server. A copy of our neural network is sent to the client and each copy is trained autonomously on the respective device. The newly formulated model weights are then delivered to the central server for aggregation and the newly formulated model is again sent to the devices. Therefore, the model present at each device will pick up the data collected from all devices and the training at the shared global server is not required. This results in the use of lower bandwidth and takes place when a connection to the central server is accessible (Fig. 1).

2.2 Anomaly Detection

Anomalies are also termed as outliers, deviants, or abnormalities. An anomaly is so different from the usual entry in a system that generates doubt of its authenticity. An anomaly can be encountered due to malfunctioning systems, cybercriminals, or just an entry that was never encountered previously in the system. Anomaly detection is the detection of fake entries in a set of online exchanges. Anomaly detection has various applications like credit card fraud detection, online payment, health monitoring, bug, or fault detection. Anomaly detection was being done with machine learning approaches as it was a big thing in the previous time, but as time advanced, more deep learning techniques were used to classify anomalies.

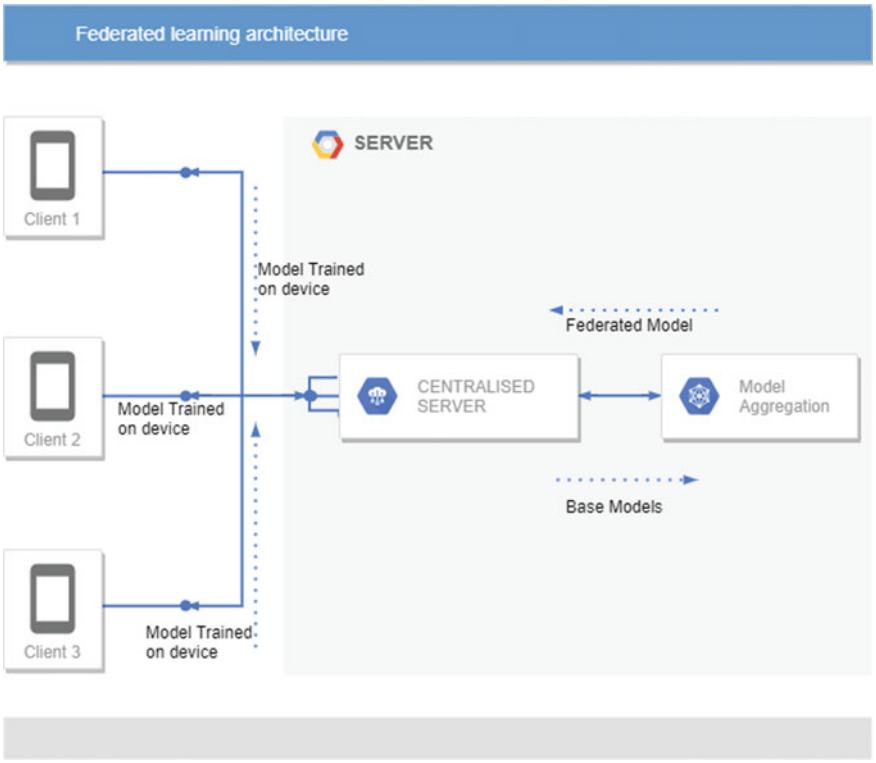

Fig. 1 Federated learning architecture

2.3 Deep Learning

Deep learning is a subset of machine learning which consists of models having multiple layers to train over data with layers of abstraction [7]. The domain of deep learning is similar to the behavior of the human brain with algorithms that resembles a neural network. Simple machine learning algorithms find it difficult to detect the patterns remotely present in large datasets whereas deep learning is found to be more productive with vast datasets, huge models, and large computations. CNNs can be applied in fields like video, audio, and images whereas RNN has the capability to work on data consisting of text and speech. Deep learning is self-sufficient to extract the features from raw data.

2.4 *Artificial Neutral Network*

An artificial neural network is a deep learning model which is fed on a vector of values; it then performs naive calculations based upon how the ANN is defined and then produces an output. The fed vector of values could be any form of data, from heterogeneous features to words used in a sentence and the output might depict a label.

The process of training feeds on known inputs and expected outputs, and alters the network on the basis of the anticipated result. This, when combined with the actual result produced by the neural network is employed to refine and reproduce the expected outputs. The network then attempts to produce an output that matches the expected behavior, such as correctly identifying the breed of a dog in an image.

The name deep learning comes from the multiple layers hidden between the input and the output layers as depicted in Fig. 2. The hidden layers are made up of neurons that carry out basic computations on the input received and pass the computed result to the subsequent layer.

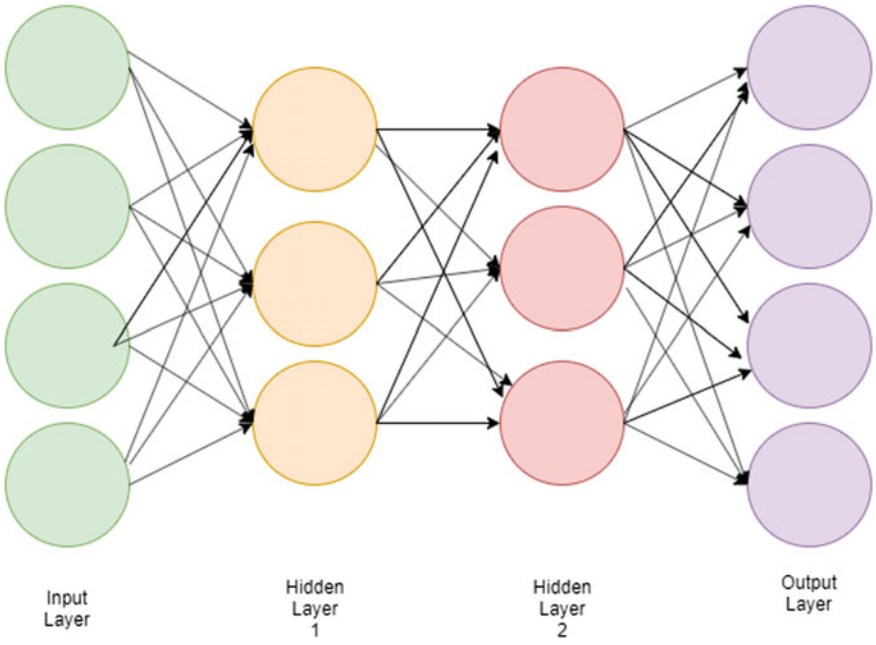

Fig. 2 Neural network model

2.5 Autoencoder

Autoencoders, alternatively known as auto-associative neural network encoders, attempt to regenerate the data fed to them as the output. A pair of symmetric neural networks forms the autoencoders. The first one does the job of compression; it will encode the input into a compressed form, while the second one will do the opposite of it and decodes the data provided to it. During the process, the middlemost layer receives the most dimensionally reduced form of the data provided initially. This will help us in recognizing the common inputs while the uncommon inputs will get the spotlight as their error will be prominent during the regeneration. The error during the regeneration would be the basis of anomaly score which will help us whether the observation is anomalous.

3 Experiment

Implementation of this paper is done on PyTorch models, and PySyft library is used for implementing the federated part of the project. Dataset was sampled in two parts in equal fractions which made our two clients for training model on different data which will be used for aggregation. The dataset consists of variables having a wide range of values which cannot be compared given its raw form. So, data is preprocessed to make it easier to compute the results and study anomaly in the dataset. We use the following formula for the normalization:

$$X = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{1}$$

Then, we create the model of our sparse autoencoder using ReLU and Tanh layers which are used to reconstruct the input points. Extra variables are deleted as it can block the necessary memory which is required for models and important variables as we are using CUDA and it has limited cache memory. Common problem while loading datasets like KDD99 [8] is that CUDA runs out of memory. So, make sure to free variables as soon as their work is complete.

After this, we hook two workers with the two given fractions of datasets and train them, respectively. Both the models are trained on different data and then aggregated using PySyft to get the federated model and then it is used for prediction.

Using MSELoss, i.e., mean squared error between every element of input and output or target, we determine that if the point is an anomaly or not. If the loss is above the set threshold up to which we consider it a normal observation, it is treated as an anomaly.

We used the machine with the following specifications:

Software: OS: Windows 10, Jupyter Notebook, Python 3, pip.

Table 1 Results of the experiment done on shuttle dataset

Model name	Training data points	Client fraction	False +ve	False −ve	F1 score
Federated model	46,463	0.4	38	40	1
Full model	46,463	NA	0	263	0.97

Hardware: CPU: i5 7300HQ, GPU: GTX 1050Ti 4 GB, RAM: 8 GB. Disk: 128 GB.

4 Result

20% of the dataset was kept for testing and the remaining was divided into two subparts which were treated as two clients to be trained separately. These clients send back the model to produce the federated model. The results showed that the merged model was able to perform better than the model trained on full dataset.

We encounter the following results after testing on federated model and model trained on full dataset shuttle unsupervised [9] (Table 1).

So, with these results, it is safe to say that by using PySyft, we got good results and comparable to the results of Schneible et al. [10]. We were able to classify all the anomalies positively.

5 Conclusion

In conclusion, we implemented federated learning with the help of PyTorch and PySyft using two datasets of Unsupervised Anomaly Detection Dataverse from Harvard Dataverse [11]. We saw that comparable results could be achieved with the different technologies used to do the same task.

References

1. H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Aguera y Arcas, *Communication-Efficient Learning on Deep Networks from Decentralized Data* (AISTATS, 2017)
2. V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 6 (2009)
3. M. Martinelli, E. Tronci, G. Dipoppa, C. Balducelli, Electric power systems anomaly detection using neural networks, in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems* (2014)
4. H. Ullah, M. Ullah, N. Conci, Real-time anomaly detection in dense crowded scenes, in *Video Surveillance and Transportation Imaging Applications* (2014)

5. A. Dau, V. Ciesielski, A. Song, Anomaly detection using replicator neural networks trained on examples of one class, in *SEAL* (2014)
6. C. David et al., A practical tutorial on autoencoders for nonlinear feature fusion: taxonomy, models, software and guidelines. Inf. Fusion **44**, 78–96 (2018)
7. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. Nature (2015)
8. G. Markus, "kdd99-unsupervised-ad.tab", Unsupervised Anomaly Detection Benchmark (2015). https://doi.org/10.7910/DVN/OPQMVF/GIPF3O, Harvard Dataverse, V1, UNF:6:WwXF9CrMJIdTvBZfZ4vpyg==[fileUNF]
9. G. Markus, "shuttle-unsupervised-ad.tab", Unsupervised Anomaly Detection Benchmark (2015). https://doi.org/10.7910/DVN/OPQMVF/VW8RDW, Harvard Dataverse, V1, UNF:6:sQatmd5Ao0CdXQULYgoDqQ==[fileUNF]
10. J. Schneible, A. Lu, Anomaly detection on the edge, in *MILCOM 2017–2017 IEEE Military Communications Conference (MILCOM)* (IEEE, 2017)
11. G. Markus, *Unsupervised Anomaly Detection Benchmark*. https://doi.org/10.7910/DVN/OPQMVF, Harvard Dataverse, V1, UNF:6:EnytiA6wCIilzHetzQQV7A==[fileUNF]