Journal of Cyber Security

Tech Science Press

DOI: 10.32604/jcs.2023.040186
Article

Seeded Transfer Learning for Enhanced Attack Trace and Effective Deception

Jalaj Pateria[1,*], Laxmi Ahuja[1] and Subhranil Som[2]

[1]Department of Amity Institute of Information Technology (AIIT), Amity University, Noida, Uttar Pradesh, 201301, India
[2]Department of Information Technology, Bhairab Ganguly College, Kolkata, West Bengal, 700056, India
*Corresponding Author: Jalaj Pateria. Email: pateria_jalaj@hotmail.com
Received: 08 March 2023; Accepted: 27 April 2023; Published: 10 August 2023

Abstract: Cyberattacks have reached their peak during COVID-19, and intruders urge to gain the upper hand in the cybersecurity battlefield, even gaining dominance. Now intruders are trying harder to elude behavior analysis techniques, which in turn gets organization security to come for a toss. This phenomenon is even more prevalent in agentless environments (IOT devices, mobile devices), where we do not have any access to edge devices and rely on packet data to predict any attack and its actors. In this paper, we shall be discussing enhancing the accuracy of anomalous behavior detection techniques for efficient threat intelligence and revamping deception using a unique machine learning model training technique termed "Seeded Transfer Learning", in this technique data is reshaped into knowledge to fit to the target domain in small capsules of information in real-time or near real-time. In this method we shall be using seeds of data or real-time small data sequences to train machine learning models along with that sustaining on the ideology of positive transfer learning techniques where previous learning will be made more effective by taking advantage of a new training set. Comprehensive experiments are done on ANN-(Artificial Neural Network) and result revels best incremental performance from ~91% to ~97%. Experiment summarized that while training model on data seeds we have achieved great accuracy with limited computing resources and time, additionally model is trained on latest attack dataset which helped in identifying attacks effectively which in turns translates to effective defense against future unknow attacks.

Keywords: Breadcrumbs; decoys; event chaining; artificial intelligence; cybersecurity; deception technology; threat intelligence; transfer learning; seeded learning

1 Introduction

With the growing network and increased Internet traffic, cybersecurity is becoming a critical battlefield, and deception technology platforms are expected that they come out of traditional point behavior analysis and upgrade themselves to learn from previous attacks or compromises withstanding knowledge of previous trainings. Hence, there is a need to bring an effective model training mechanism

to train behavior analysis and anomaly detection algorithms periodically. This could be achieved by training them on variable scenarios, datasets in the form of data seeds and are fed to the model which are near real in nature, so that models are up to date as per latest attack trends as discussed in reference [1].

In this, we shall be using two specific machine learning concepts, "transfer learning" which works on principle of domain adaptation that specifies utilizing previous rigorously trained model and retaining the knowledge gained from previous trainings topped with "seeded learning" which specifies training model on small datasets to gain maximum advantage on anomaly detection and learning patterns to enhance attack trace and effective deception as summarized in references [2,3].

The core objective of the proposed experiment is to build a network protection mechanism incrementally on limited data, as new anomalous patterns are not a frequently occurring. Model training shall be done on a small sample of network data termed seeds, which is new in nature and sufficient for enhanced network protection. Iterations of training happen over time to incrementally update the model with input data from classes that it failed to detect, advancing its protective capabilities. Attack data, which is limited, is fed to ML models for training, after the initial round of training, the model will be again reloaded and retrained on a new set of attack data as highlighted in Fig. 1. This helps in uncovering zero-day vulnerabilities with a model capable enough to look for any new behavior's anomaly. This activity gives our deception frameworks an extra advantage to detect intrusions and act on them quickly.

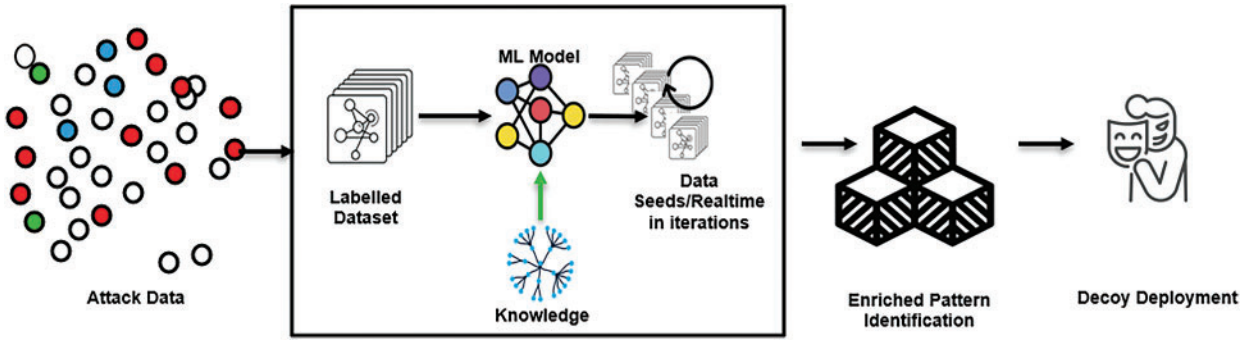

Figure 1: System architecture of deception technology with seeded transfer leaning

The next section of the paper considers a discussion of Research motivation. Section 3 contains the literature review. Methodology is showcased in the Section 4. Dataset analysis & proposed experiment and method can be referred in Sections 5 & 6. Result analysis and conclusion can be derived from Sections 7 & 8. Future scope of work is showcased in Section 9.

2 Research Motivation

Behavior analysis or anomaly detection models, once trained on huge amounts of data, need to be retrained to make them relevant in the attack space. To deal with zero-day vulnerabilities, it is important to overcome the isolated learning barrier, which doesn't utilize knowledge gained in the past for future learning and solve related ones. With seeded transfer learning, previously learned attack detection behavior can be topped up with newer attacks and compromises as discussed in Fig. 2. Now, to cover up a data scarcity gap which is not available on the fly for traditional model training due to the

limited occurrence of attack sequences and attack data traces, we need a technique that can transfer knowledge from one model to another or from one training iteration to another. Transfer learning encompasses knowledge (features, biases, weights, activation, layers, etc.) from former trained models and is used to train new models or upgrade existing ones, which knocks out problems like having less data for the newer training activity as discussed in Fig. 2 and references [2,4–6].

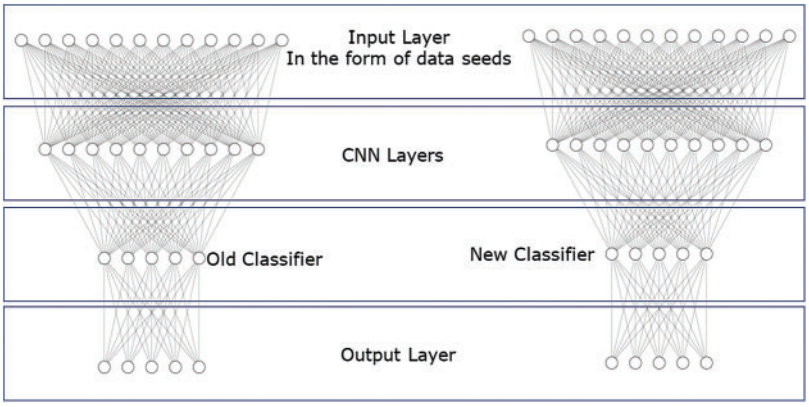

Figure 2: Seeded transfer learning & domain adaptation updating classifier approach

This paper focuses on improving enterprise deception strategy by analyzing and training existing knowledge of behavior analysis models with small data named seeds. Majorly consists of the latest attack reconnaissance activities on the network, domain adaptation, and making models relevant to cater to zero-day attacks, vulnerabilities, and new attack patterns, thereby devising an effective deception strategy as discussed in references [7,8].

Below are a couple of benefits, not limited to:

- Effective controls and measures can be placed to protect enterprises from zero-day attacks and vulnerabilities. With seeded transfer learning, behavior analysis models for attack or anomaly detection matures over iterations which happens over short time intervals on small data packet, but classical training is computationally expensive and happens from scratch on huge data as highlighted in reference [9].
- Due to greater visibility to network uncovering of various compromise technique patterns with behavior analysis algorithms go simple, hence management of defense is effective, and a revamped deception strategy can be crafted accordingly. Which previously was not practical extended wait time for new attack data to flourish and retaining of previous knowledge was not tranquil as per reference [10].
- Now that models are well updated as per new threat patterns, readiness, and the ability to react to and recover from security events can be strengthened to reinforce the current security posture by identifying possible security gaps and taking action to eliminate those gaps as highlighted in reference [10].

3 Literature Review

Lot of research is happening from a deception technology perspective, and stress is given to the advance prediction of behavior deviation efficiently and with speed to have an effective defense against a zero-day attack, which is one of the toughest problems to solve in the cybersecurity space. In the past, behavior's deviations were experimented on using neural networks, genetic algorithms,

SVM, and multilayer perceptron neural networks, which could be applied in behavior anomaly detection to recognize and classify types of attacks as presented in reference [11]. Traces of study have been conducted in the fields of transfer learning and domain adaptation. Implementation of seeded transfer learning accompanies domain adaptation and is getting the attention of the machine learning community. Wu et al. proposed a concept ConvNet which can be applied to network intrusion detection which using transfer learning for the network intrusion detection which contains two concatenated ConvNets were learning a base dataset and transferring the learned knowledge to the learning of the target dataset [12]. Li et al. experimented on density-based clustering (DBSCAN) to convert features into categorical as hybrid architecture representations as per reference [13]. Osama et al. discussed on various attack sequences on IOT devices and its segregation using various federated learning techniques as highlighted in reference [14]. Mathew et al. showcased a case study on domain adaptation where transferred information fed to multiply connected layers for intrusion detection as experimented in reference [15]. Zhao et al. talked on using transfer learning on various algorithms decision trees, random forests, KNN which promotes automatic co-relation and denied the requirement on using labelled dataset and helps in classifying unknown attacks as researched in reference [16]. Long et al. talked about domain transfer on different sizes for source domain data; different noise levels to drive generic algorithm and domain adaptation on different noise levels dataset, verify if transfer learning is possible on unlabeled data as conversed in reference [17]. Partial domain transfer was explored by Deng et al. which transfer of knowledge between two disintegrated domains as experimented in reference [18]. Transfer Learning is well discussed by Agarwal et al. which highlights positive transfer and negative transfer along with its effects on accomplishment of learning in target domain [19]. Javaid et al. discussed about approach utilizing Deep Learning model in a classical way as analyzed in reference [20]. DOS attacks are studied in specific discusses on deep learning framework using patterns matching and classifier algorithms is discussed by Lin as explored in reference [21]. Further research on domain adaptation discussed by Pan et al. which states that deep neural networks can enhance the invariance using unified framework, Despite knowing the fact that domain adaptation and transfer learning work on a similar set of principles for transferring knowledge across various domains, it is important to note that transfer learning may not distinguish the target domain from the source domain as understood from reference [22]. It has also been highlighted in the studies that seeded transfer learning provides an advantage in terms of training time and computational resources and conquers the problem of data availability in small packets or seeds. Implementing the transfer learning weighted loss function is a very usual problem that any implementor faces, and research is going on to handle the issue of class imbalance. Even a couple of research papers are talking about "transferred information from the initial inception model that has been fed to various fully connected layers with dropped data to achieve better accuracy".

Concluding, all the previous studies either demonstrates transfer learning or seeded learning on various cybersecurity aspects but have not touched upon behavior analysis algorithms mainly inclined to a combination of seeded transfer learning and domain adaptation logic in deception technology space, which is a huge conceptual disintegration. Besides having previous gaps, negative transfer is not dealt properly wherein data drift and model drift is rarely discussed throughout the transfer learning lifecycle. In the proposed concept we are targeting to use existing behavior analysis model which belongs to the near similar domain and will get them adapted to trace new attack trends by training them incrementally in data seeds to capture latest attack trends and secure enterprise backbone effectively as examined by Long et al. in reference [23].

4 Methodology

Threats appears in different dimensions; it can be malicious or accidental. Analysis of Interactions of external agents with spread across the network, hosted on various services and ports assists in finding anomalous behavior by connecting attack series and sequences to get an attack coverage. Any interaction when evaluated carefully by validating it against multiple attack surfaces assist in bring up efficient deception thereby robust Security Framework and intelligent deception. In the experiment we are considering data dimensions when Training dataset which will be used as seeds for Seeded transfer learning is having different property from the base training dataset and it is small or streaming in nature as highlighted in references [2,18].

Each function f, mapping to an input x f(x). Learning produced from previous iteration fed to next iteration. Transmission process to be executed till nth iteration as discussed in Fig. 3. Usually in these types of experiments overfitting might be a concern, as highest contributing features are not same and neural network might use initial few layers as highlighted by Poornachandra in reference [24]. In that case there is a need to remove all the pre-trained layer from the neural network and add pretrained layers with new layers to according to new dataset with random weights and freezing weights from new and old layers. And then train the network with fully connected layers. Due diligence has been taken care to validate the model performance and evaluate the performance before saving it based on the transfer learning results, if learning is positive transferred model shall be saved to be used for next iteration otherwise discard the model training in case learning is negative transferred as explained in Fig. 4 and references [25,26].

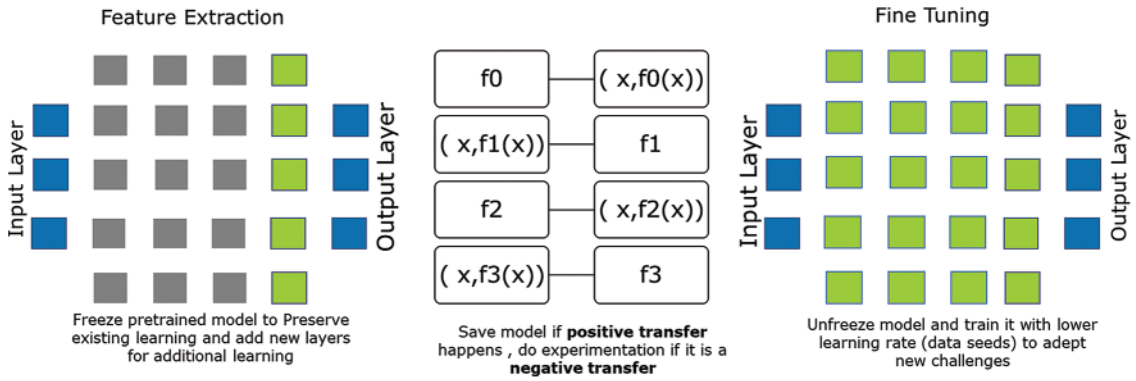

Figure 3: Model function for knowledge adaptation retaining knowledge and finetuning

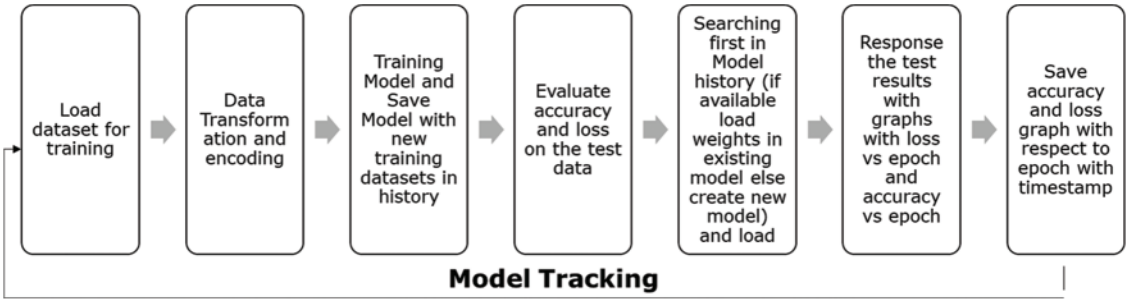

Figure 4: Seeded transfer incremental learning lifecycle

5 Dataset Analysis

The dataset contains records of the traffic experienced by an intrusion detection network or defense; the ghosts of the traffic categories list countered by a real IDS remain along with a trace of its existence. Out of total 43 features, 41 features are referred to input network traffic and rest two labels are signifying if interaction to IDS is a normal or attack and severity of the moving traffic (Table 1). Analyzing can help enterprises enable robust security postures, provide greater visibility to the threat landscape, and enable a deception strategy. This goldmine of information, when related, provides greater visibility to the intrusion that is going to happen and helps in immunizing the defense mechanism for any attack that tracks similar behavior and the direction of progression.

Table 1: Parameters and data statistics of input dataset

Dataset	Number of records					
	Total	Normal	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449 (53%)	9234 (37%)	2289 (9.16%)	11 (0.04%)	209 (0.8%)
KDDTrain+	125973	67343 (53%)	45927 (37%)	11656 (9.11%)	52 (0.04%)	995 (0.85%)
KDDTest+	22544	9711 (43%)	7458 (33%)	2421 (11%)	200 (0.9%)	2654 (12.1%)

Features in a dataset represent measurable pieces of data that can be used for analysis. These are individual or independent variables that act as inputs in machine learning models and shall be used to perform prediction or classification operations as highlighted in Table 1. In the dataset we have used, we have 43 features.

Target column or variable historical data to be analyzed is co-related to uncover patterns and relationships between other features of the dataset and the target; below are the target columns used to contribute to the seeded learning of the Attack dataset:

A. **Denial of Service:** It is a type of attack that targets blocking incoming and outgoing traffic from the target system. Intrusion detection systems face abnormal traffic, which they are not able to handle and shut down to protect themselves. In our analysis, we shall be using the below DOS subclasses: [‘processtable’, ‘apache2’, ‘mailbomb’, ‘back’, ‘land’, ‘neptune’, ‘pod’, ’smurf’, ‘teardrop’, ‘udpstorm’, ‘worm’].

B. **R2L (Remote to Local):** To get access to a remote machine and use it locally in the network, the intruder opts for the below techniques. [‘httptunnel’, ‘ftp_write’, ‘warezclient’, ‘multihop’, ‘imap’, ‘named’, ‘phf’, ‘snmpgetattack’, ‘snmpguess’, ‘guess_passwd’, ‘sendmail’, ‘spy’, ‘warezmaster’, ‘xlock’, ‘xsnoop’]:

C. **Probe:** With probe, an attacker tries to get network information and compromise important information. Techniques what intruder uses [‘ipsweep’, ‘mscan’, ‘nmap’, ‘portsweep’, ‘saint’, ‘satan’]:

D. **U2R (User to Root):** Gain access to the root account of the user and then exploit system vulnerabilities with modified elevated access. [‘buffer_overflow’, ‘loadmodule’, ‘perl’, ’ps’, ‘rootkit’, ‘sqlattack’, ‘xterm’]:

Categorical column: This data generally has a limited number of possible values. For our purpose, we have considered protocol type, service, and flag.

6 Proposed Experiment and Method

The proposed method uses a neural network and a small sample from the attack dataset to transform. It uses limited attack samples as seeds to initiate the transfer of source knowledge and domain adaptation. A series of activities is conducted to develop an experiment pipeline, wherein stress is given to ingest data that is in real-time or near real-time as specified in Fig. 4. The scope of this paper talks about doing comprehensive training experiments on the NSL-KDD dataset, iterating through multiple obtained results, doing comparisons against similar models trained with a knowledge source or deep learned features, and validating the iterations against accuracy, EPOCH, and loss.

6.1 Feature Engineering

The feature engineering process is associated with a series of preprocessing steps executed to transform raw data into features that can be used in machine learning algorithms. During the feature engineering process, the predictor variables with the highest weight are created and selected for the predictive model and can be referred from Fig. 5. Feature engineering includes feature creation, transformations, selection, and extraction of data features. A series of steps were followed to do effective feature selection from the attack dataset. When feature engineering is applied to an attack dataset that is either a text file or comma-separated, header names should not be considered. Get the dataset loaded with the feature name and then change the attack labels to their respective attack classes. Fig. 5 suggests feature importance using Extra Trees Classifier which is and ensemble technique of machine learning that uses decision trees and used by Auto-ML with a capability to build multiple decision trees over dataset and for standardization in feature scaling, numeric values must be selected.

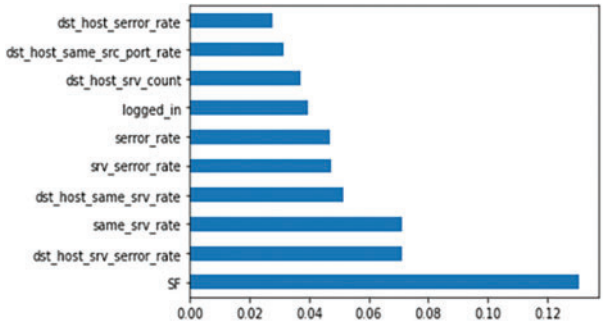

Figure 5: Feature importance (extra trees classifier)

If scalers already exist, then load the scaler. Otherwise, save the scaler of the dataset after fitting it with training data. To transform the target column, label encoding is used, and one hot encoding is used for categorical columns (protocol type, service, and flag) as interpreted from Table 2. Then we normalize the dataset and drop categorical column to make this ready for model training.

6.2 Model Architecture

Model Architecture Post-feature engineering, we shall be using a feedforward artificial neural network (ANN) named Multilayer Perceptron that classifies and categorizes any attack instance. Initial model training to be done on a dataset to obtain the knowledge, and from the next iteration on, a seeded dataset is being used for training purposes. If a model exists, then there is a need to load the previous model and then save it to the model history directory as understood from reference [12]. This procedure translates to seeded transfer learning, where small data seeds are used to train the data and save the model in an incremental way by updating model knowledge. To train a model with the dataset

on a sequential model, first initialize the model sequentially and add dense units (50) and activation as relu for the input layer. Then add dense units 5 and activation as softmax for the output layer. Then, compile with loss categorical_crossentropy, the optimizer as Adam, and metrics as accuracy and can be referred from Fig. 6 and Table 3. Model Summary highlights important information regarding model training parameters layers and params and can be understood from Table 3.

Table 2: Dataset categorical features description

Categorical column	Description	Categories examples (not limited to)
protocol_categorical_type	Highlights set of rules to transfer data between different devices in a network	'icmp', 'udp', 'tcp'
service_categorical_list	Highlights set of services to transfer data between different devices in a network	'ftp_data', 'other', 'private', 'http', 'remote_job', 'bgp', 'name', 'netbios_ns', 'eco_i', 'mtp', 'telnet', 'finger', 'uucp', 'klogin'
flag_categorical_list	Describes features of each connection instance.	'SF', 'S3', 'S2', 'S0', 'RSTOS0', 'REJ', 'RSTR', 'SH', 'RSTO', 'S1', 'OTH'

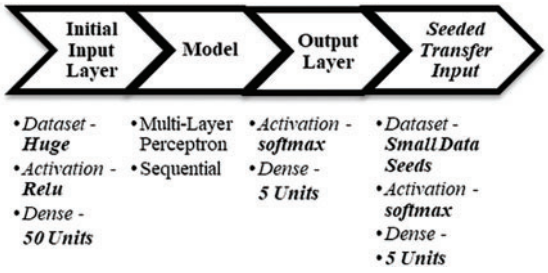

Figure 6: Parameters and Cycle for the proposed experiment

Table 3: Model training summary and statistics

"Model: "sequential_1"

Layer (type) Output Shape Param #
==========
dense_1 (Dense) (None, 50) 6150
dense_2 (Dense) (None, 5) 255
==========
Total params: 6,405
Trainable params: 6,405
Non-trainable params: 0

6.3 Model Architecture

The initial step is to load the pre-trained model and then compile it with loss categorical_crossentropy, which is a loss function based on formula as per Fig. 7. that is used in multi-class classification problems considering each class is assigned a unique integer value. Then it calculates the average difference between the probability distributions of all classes, and the score is minimized. We should be using the Adam optimizer, which is adaptive in nature and uses the stochastic gradient descent method with a best-in-class accuracy of 99.2%. The metrics used are accuracy. Batch size and epoch should be selected suitably so that the model is not overfitted as investigated in reference [27].

$$.LOSS = -\sum_{i=1}^{Output\ Size} yi.\log(y\hat{}i)$$

Where, yi *denotes real Value or truth level*

y^{hat}(y *hat*) *denotes predicted probablity of the classifier*

Figure 7: Categorical cross-entropy loss function

If you encounter overfitting, then dropout weight constraint, activity regularization, or early stopping should be done, sample code can be referred from Fig. 8. While training with the new dataset, if the performance of the new model remains constant without any improvement, then ML OPS should come into play. If new data consists of categories that are not inherently present in a previously trained dataset or pipeline, MLOps should be triggered as understood from reference [28].

```
if exists(model_path+"model1.h5"):
    mlp1 = load_model(model_path+"model1.h5")
    model_history= model_history_path +str(datetime.now())[0:19].replace(":","_").replace(" ","_")+"model.h5"
    mlp1.save(model_history)
    model_record_path=model_history.replace(model_history_path,"")
    # defining loss function, optimizer, metrics and then compiling model
    mlp1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    history = mlp1.fit(X_train, y_train, epochs=20, batch_size=1000,validation_split=0.2)
    mlp1.save(model_path+"model1.h5")
    graph_record_path_acc=self.generate_acc_graphs(history)
    graph_record_path_loss=self.generate_loss_graphs(history)
```

Figure 8: Training pre-existing model reference code

Model drift is the usual phenomena and result findings should be validated against various drift aspects like Sudden, Gradual, Incremental, or reoccurring drift and then take necessary tuning or redeployment actions to bring that to track.

7 Result Finding

Evaluation and seeded transfer learning on attack data have been accomplished. As a part of this experiment, we have divided the data into seven subsets: the first subset for initial training data and six smaller subsets. A total of seven iterations were executed on the sample of 10,000 unbiased and no-skewed observations having all the labels of normal traffic flow and compromised state. Out of these seven iterations, the first is an initial training, and the rest are seeded learning iterations. Model performance is evaluated on accuracy against epoch, which represents model performance in terms of comparable performance on both training and validation attack datasets as summarized in Table 4. If the parallel plots start to go away from each other consistently, it means necessary pointers are gained to validate if the model is overfitting, and we should stop seeded transfer learning at an earlier EPOCH phase and get into MLOPs and remove all the pre-trained network layers and do retraining again as per new random weights the results can be reviewed from Figs. 10–12. Another aspect of evaluation is to look for a decaying learning rate by evaluating against loss and epoch, which works on the loss

function and plots across iterations to give loss on a subset and can be understood from Table 4. To do a granular study of results, we must understand the functioning, relevance, and relation of "accuracy *vs*. epoch" and "loss *vs*. epoch".

Table 4: Result analysis of seeded transfer learning iterations

Round	Train data	Test data	Loss	Accuracy (%)	Batch size	Epoch
Initial training	(10000, 42)	(5972, 42)	0.643903196	91.51%	6000	20
Round 1	(9999, 42)	(5972, 42)	0.245419323	94.78%	6000	20
Round 2	(9999, 42)	(5972, 42)	0.144316897	96.38%	6000	20
Round 3	(9999, 42)	(5972, 42)	0.11116372	96.84%	6000	20
Round 4	(9999, 42)	(5972, 42)	0.089884989	97.39%	6000	20
Round 5	(9999, 42)	(5972, 42)	0.074087664	97.52%	6000	20
Round 6	(9999, 42)	(5972, 42)	0.064278908	97.69%	6000	20

"**Epoch**" indicates the pass of a dataset through the algorithm; forward and backward passes are considered as a single pass.

The "**accuracy**" of a model is determined after the model parameters are fixed and no learning is occurring.

"**Loss**" is a value that represents the sum of errors in our model. It measures how well (or badly) our model is doing. The lower the loss, the better. Highlights of "accuracy *vs*. epoch" showcases how the accuracy of a model is increasing against each epoch.

"**Loss *vs*. epoch**" showcases the summation of errors in our model by dividing the running loss by the number of epochs.

The accuracy chart and table showcase the seeding transfer learning phase goes well till third iteration post that graph started to depart from each other once reached to sixth iteration features the plots are substantially far enough away, giving a sign to stop further training and initiate artificial implantation on neural network layers. This experiment has also presented the relation explaining a linkage of the number of training records and the accuracy.

The accuracy of the experiment is on the uptrend, with a 91.51% prediction result in the initial iteration. Subsequent training until iteration 6 represented 97.69% accuracy and can be understood from Figs. 10–12. With the results, it is understood that models can be fed with small data seeds and can proceed with the model training in near real time, and the proposed data pipeline highlights that in case of any data drift where model performance goes down with regular analysis, we can avoid degrading model performance and can apply various model performance enhancement strategies. Result analysis has also led us to the conclusion that transfer learning will become ineffective when high-level features of the operating domain are not differentiated correctly with the bottom layers as highlighted in Fig. 9. For example, a pre-trained model might correctly identify the current state of compromise but not whether that compromise is an actual attack or not. In this case, we can still reutilize the deep learning network by utilizing low-level layers if the network is pre-trained instead of the high-level features due to the lack of correct prediction by high-level layers. Effort should be made to retrain more layers of the model. Initializing the network with pre-trained weights helps

attain better performance than using random weights. Fewer times, removal of some layers from the pre-trained model is needed to reduce the number of trainable parameters, which can result in overfitting, and we need to deal with those cases carefully and same can be reviewed from Fig. 9 and reference [22].

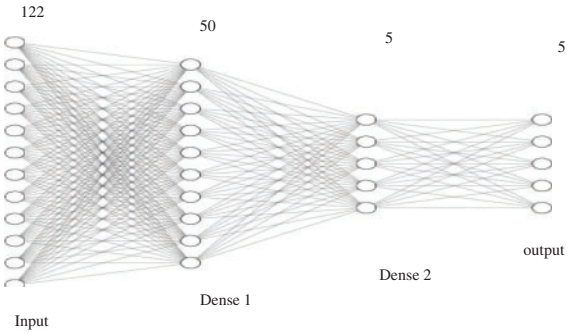

Figure 9: Model layers, parameters and summary

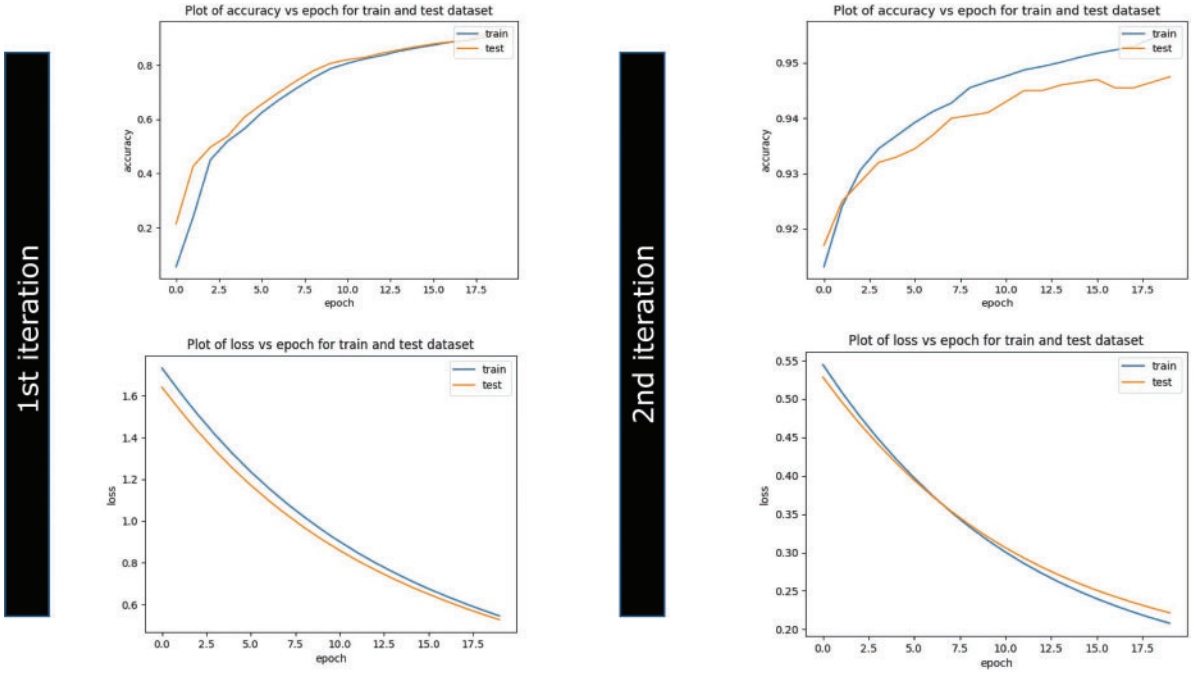

Figure 10: Iteration 1 & 2 plots (accuracy *vs*. epoch) (loss *vs*. epoch)

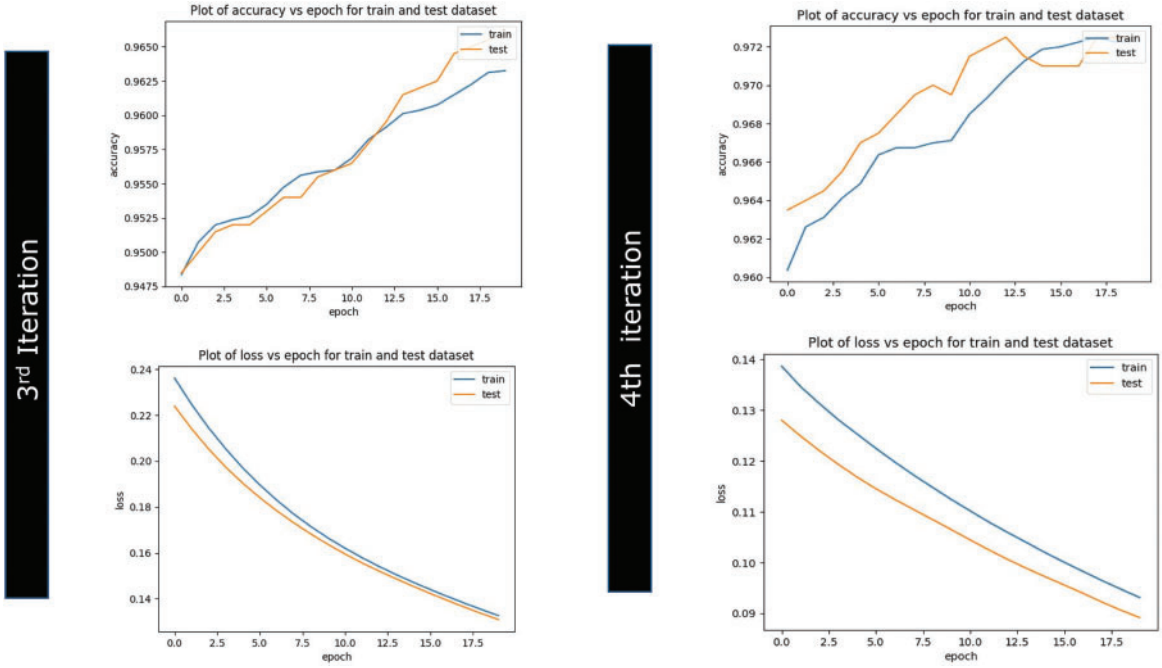

Figure 11: Iteration 3 & 4 plots (accuracy *vs*. epoch) (loss *vs*. epoch)

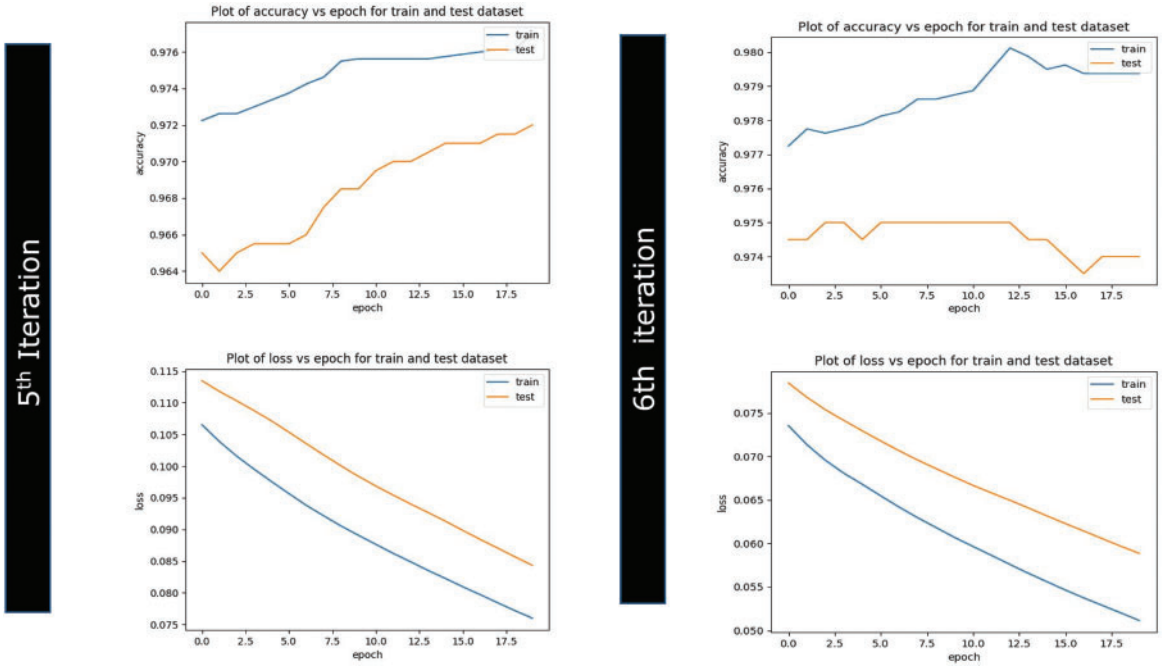

Figure 12: Iteration 5 & 6 plots (accuracy *vs*. epoch) (loss *vs*. epoch)

Aforesaid comparison showcases that performance of a model stands outstanding when trained via seeded transfer learning process and there is a uptrend in accuracy for couple of iterations before in attains negative transfer or no transfer state. But to understand how progressive this thought is in

comparison to Tradition model; effectiveness is validated via experiment where accuracy is captured when training is done on sample size of 42000 with Traditional mode training procedure against training using seeded transfer learning in sample size of 6000. Results clearly articulates that when training done on huge data, we get comparatively good accuracy in comparison to seeded learning Initial and First iteration. Post that seeded Transfer learning mechanism has got comparatively good accuracy in subsequent rounds and same can be articulated from Fig. 13.

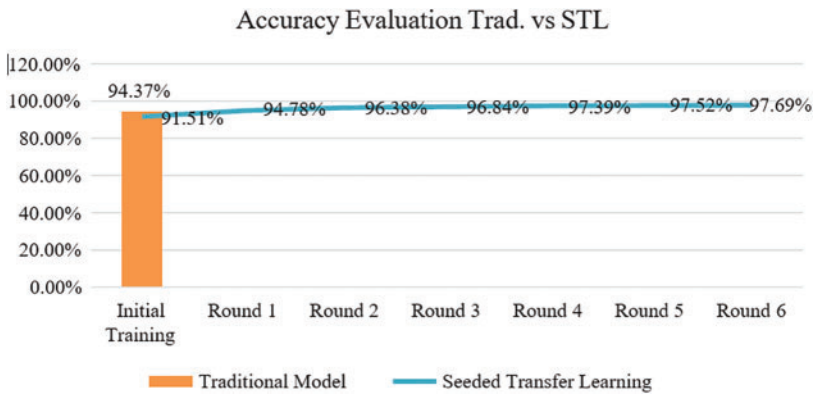

Figure 13: Comparison between traditional *vs*. seeded transfer learning

8 Conclusion

Cybersecurity is an unending battle, and defenses need to be updated beforehand before any attack or compromise starts showing its traces in the system. Zero-day vulnerabilities and latest attack trends need to be well defined in behavior analysis machine learning models, which help uncover any behavior deviation of the intruder as our intelligence system is updated as per the latest attack sequences. Additionally, static model training on full data is lacking in adaptability due to the dynamic nature of attack data over time. Additionally, availability of training (Attack) data as per requirement if is not available in abundance, there is a need to migrate from traditional model training approach where huge data to be used and might take long time for data reconciliation and huge computing power for training those models. This waiting period, during which organizations are looking forward to huge amounts of training data, can cause a security issue. Additionally, as systems can detect any threat well in advance because they are trained on the latest attack data seeds, deception strategies can be well adapted and crafted as per the new attack patterns and enhanced as per changing requirements.

9 Future Research

Intrusion Detection Systems have been used in organizations to dynamically screen networks and provide threat intelligence to defend against future attacks, even deception technology is playing a great role and added an additional defense layer to the threat landscape to bluff attackers and get increased attacker dwelling time to network This paper focused on increasing accuracy of those behavior analysis models using seeded transfer learning, additional study to be done to make this concept even more progressive by tuning and experimenting on unlabeled dataset and latent parameters. Additionally, here we have selected source and target domains that are related. The problem with negative transfer is still a problem to address when source and target domains are not related to each other in some

sense negative transfer and non-learning scenarios could be an area of interest for future researchers as addressed in references [19,23].

Moreover, any new advancement comes with a couple of demerits, and when using seeded transfer learning, training data ingestion is done to model from various sources and users, and various transactions come up with some security issues. To secure the transactions, there is a need to authenticate the source data and build consensus-based voting to validate if the data pushed to the model is authentic or not, and on that basis, train and retrain transactions to be accepted or declined. This shall be addressed as the future scope of work for upcoming research in the segment. So, to summarize the context of continuous improvement for future scientific research, researchers should pay attention to the application of various consensus-based algorithms to even fine-tune research and analysis to stay ahead of the intruders and effectively defend against various cybersecurity incidents as indicated in references [29,30].

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. W. Tzeyoung, "Tools intrusion detection systems. Information assurance tools report. Sixth edition, Distribution Statement A EXCELLENCE", Information Assurance Technology Analysis Center (IATAC), 2009.

[2] A. Sallam, "Early detection of glaucoma using transfer learning from pre-trained CNN models," in *2021 Int. Conf. of Technology, Science and Administration (ICTSA)*, pp. 1–5, 2021. https://doi.org/10.1109/ICTSA52017.2021.9406522

[3] C. Constantinides, S. Shiaeles, B. Ghita and N. Kolokotronis, "A novel online incremental learning intrusion prevention system," in *2019 10th IFIP Int. Conf. on New Technologies, Mobility and Security (NTMS)*, pp. 1–6, 2019. https://doi.org/10.1109/NTMS.2019.8763842

[4] S. M. Salaken, A. Khosravi, T. Nguyen and S. Nahavandi, "Seeded transfer learning for regression problems with deep learning," *Expert Systems with Applications*, vol. 115, no. 9, pp. 565–577, 2019. https://doi.org/10.1016/j.eswa.2018.08.041

[5] J. Esmaily, R. Moradinezhad and J. Ghasemi, "Intrusion detection system based on multi-layer perceptron neural networks and decision tree," in *2015 7th Conf. on Information and Knowledge Technology (IKT)*, pp. 1–5, 2015. https://doi.org/10.1109/IKT.2015.7288736

[6] A. Saeed, F. D. Salim, T. Ozcelebi and J. Lukkien, "Federated self-supervised learning of multisensor representations for embedded intelligence," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 1030–1040, 2021. https://doi.org/10.1109/JIOT.2020.3009358

[7] A. Sarhan, J. Rokne, R. Alhajj and A. Crichton, "Transfer learning through weighted loss function and group normalization for vessel segmentation from retinal images," in *2020 25th Int. Conf. on Pattern Recognition (ICPR)*, pp. 9211–9218, 2021. https://doi.org/10.1109/ICPR48806.2021.9412378

[8] R. J. Araújo, J. S. Cardoso and H. P. Oliveira, "Topological similarity index and loss function for blood vessel segmentation," arXiv preprint arXiv:2107.14531, 2021.

[9] N. Van Huynh, D. T. Hoang, D. N. Nguyen and E. Dutkiewicz, "DeepFake: Deep dueling-based deception strategy to defeat reactive jammers," *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6898–6914, 2021. https://doi.org/10.1109/TWC.2021.3078439

[10] J. Kim, J. Nam, S. Lee, V. Yegneswaran, P. Porras *et al.,* "BottleNet: Hiding network bottlenecks using SDN-based topology deception," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3138–3153, 2021. https://doi.org/10.1109/TIFS.2021.3075845

[11] M. W. Gardner and S. R. Dorling, "Artificial neural networks (the multilayer perceptron)—A review of applications in the atmospheric sciences," *Atmospheric Environment*, vol. 32, no. 14–15, pp. 2627–2636, 1998. https://doi.org/10.1016/S1352-2310(97)00447-0

[12] P. Wu, H. Guo and R. Buckland, "A transfer learning approach for network intrusion detection," in *2019 IEEE 4th Int. Conf. on Big Data Analytics (ICBDA)*, pp. 281–285, 2019. https://doi.org/10.1109/ICBDA.2019.8713213

[13] Z. P. Li, Z. Qin and P. B. Shen, "Intrusion detection via wide and deep model," in *Artificial Neural Networks and Machine Learning–ICANN 2019: Text and Time Series*, Berlin, German: Springer Science and Business Media LLC, pp. 717–730, 2019.

[14] O. Shahid, V. Mothukuri, S. Pouriyeh, R. M. Parizi and H. Shahriar, "Detecting network attacks using federated learning for IoT devices," in *2021 IEEE 29th Int. Conf. on Network Protocols (ICNP)*, Dallas, TX, USA, pp. 1–6, 2021. https://doi.org/10.1109/ICNP52444.2021.9651915

[15] A. Mathew, J. Mathew, M. Govind and A. Mooppan, "An improved transfer learning approach for intrusion detection," *Procedia Computer Science*, vol. 115, pp. 251–257, 2017. https://doi.org/10.1016/j.procs.2017.09.132

[16] J. Zhao, S. Shetty and J. Pan, "Transfer learning for detecting unknown network attacks," *EURASIP Journal on Information Security*, vol. 2019, no. 1, 2019. https://doi.org/10.1186/s13635-019-0084-4

[17] B. Long, S. Lamkhede, S. Vadrevu, Y. Zhang and B. Tseng, "A risk minimization framework for domain adaptation," in *Int. Conf. on Information and Knowledge Management*, pp. 1347–1356, 2009. https://doi.org/10.1145/1645953.1646123

[18] Y. Deng, D. Huang, S. Du, G. Li, C. Zhao *et al.,* "Double-layer attention based adversarial network for partial transfer learning in machinery fault diagnosis," *Computers in Industry*, vol. 127, pp. 10339, 2021. https://doi.org/10.1016/j.compind.2021.103399

[19] N. Agarwal, A. Sondhi, K. Chopra and G. Singh, "Transfer learning: Survey and classification," in *Smart Innovations in Communication and Computational Sciences*. Singapore: Springer, pp. 145–155, 2021. https://doi.org/10.1007/978-981-15-5345-5_13

[20] A. Javaid, Q. Niyaz, W. Sun and M. Alam, "A deep learning approach for network intrusion detection system," in *Proc. of the 9th EAI Int. Conf. on Bio-Inspired Information and Communications Technologies*, New York, USA, pp. 21–26, 2016.

[21] D. Lin, "Network intrusion detection and mitigation against denial of service attack," *Technical Report MS-CIS-13-04*, Department of Computer and Information Science Technical, University of Pennsylvania, 2013.

[22] S. J. Pan, I. W. Tsang, J. T. Kwok and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2011.

[23] M. Long, J. Wang, Y. Cao, J. Sun and P. S. Yu, "Deep learning of transferable representation for scalable domain adaptation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2027–2040, 2016. https://doi.org/10.1109/TKDE.2016.2554549

[24] S. Poornachandra, "Artificial neural networks with TensorFlow 2," in *Neural Networks for Regression*, 1st ed., Berlin: Springer, pp. 189–230, 2021.

[25] T. Wisanwanichthan and M. Thammawichai, "A double-layered hybrid approach for network intrusion detection system using combined Naive Bayes and SVM," *IEEE Access*, vol. 9, pp. 138432–138450, 2021. https://doi.org/10.1109/ACCESS.2021.3118573

[26] X. Shi, Q. Liu, W. Fan and P. S. Yu, "Transfer across completely different feature spaces via spectral embedding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 906–918, 2013. https://doi.org/10.1109/TKDE.2011.252

[27] K. B. Rabindra, R. Priyadarshini and N. Dash, "A meta-heuristic model for data classification using target optimization," *International Journal of Applied Metaheuristic Computing (IJAMC)*, vol. 8, no. 3, pp. 24–36, IGI Global, 2017.

[28] L. Wen, L. Gao and X. Li, "A new deep transfer learning based on sparse auto-encoder for fault diagnosis," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 136–144, 2019. https://doi.org/10.1109/TSMC.2017.2754287

[29] X. Chen, J. Ji, C. Luo, W. Liao and P. Li, "When machine learning meets blockchain: A decentralized, privacy-preserving and secure design," in *2018 IEEE Int. Conf. on Big Data (Big Data)*, Seattle, WA, USA, IEEE, 2018.

[30] Y. Liu, F. R. Yu, X. Li, H. Ji and V. C. M. Leung, "Blockchain and machine learning for communications and networking systems," *IEEE Communications Surveys & Tutorials*, vol. 22, pp. 2–1431, 2020.