Improving User Trust on Deep Neural Networks based Intrusion Detection Systems

Kasun Amarasinghe, Milos Manic
Virginia Commonwealth University, Richmond, Virginia, USA
amarasinghek@vcu.edu, misko@ieee.org

***Abstract*—Deep Neural Networks based intrusion detection systems (DNN-IDS) have proven to be effective. However, in domains like critical infrastructure security, user trust on the DNN-IDS is imperative and high accuracy isn't sufficient. The black-box nature of DNNs hinders transparency of the DNN-IDS, which is necessary for building trust. The main objective of this work is to improve user trust by improving transparency of the DNN-IDS by making it more communicative. This paper presents a methodology to generate offline and online feedback to the user on the decision making process of the DNN-IDS. Offline, the user is reported the input features that are most relevant in detecting each type of intrusion by the trained DNN-IDS. Online, for each detection, the user is reported the inputs features that contributed most to the detection. The presented method was implemented on the KDD-NSL dataset with a multi-layer perceptron (MLP) based DNN-IDS. Binary and multi-class classification was carried out on the dataset. Further, several DNN-IDS architectures with different depth were tested to study the factors that drive classification. It was observed that despite showing very similar accuracy results, the factors that drove the decisions were different across architectures. This evidences that the qualitative analysis that is enabled through reporting relevant input features is important for the user to make a more informed decision in choosing a DNN-IDS. This online and offline feedback leads to improving the transparency of the DNN-IDS and helps build trust prior to and during deployment.**

Keywords— Intrusion Detection, Deep Learning; Deep Neural Networks; Explainable AI; Layer wise Relevance Propagation; Anomaly Detection;

I. INTRODUCTION

Modern industrial systems including critical infrastructure systems are heavily reliant on connectivity and the ability to seamlessly connect physical and computing resources [1], [2]. In addition, with the advent of Internet-of-Things (IoT) there has been an explosive growth in connectivity among our day-to-day household items ranging from smartphones to thermostats and kitchen appliances [3], [4]. While these technologies enhance what we can accomplish at our fingertips and increase the efficiency of industrial systems and critical infrastructure, they exposes all these systems to various types of cyber threats. Malicious cyber-attacks on these systems, if successful, could lead to catastrophic events. Therefore, ensuring the securing the cyber networks is a prime concern in the modern world.

Intrusion Detection Systems (IDS) have become essential components in cyber networks. The role of an IDS is to detect unauthorized use, misuse and abuse of the computer network by insiders or outsiders [5]. Machine Learning (ML) based IDS systems based algorithms such as Support Vector Machines and decision trees have been popular in the literature [4], [6]. In the recent past, Deep Neural Networks have revolutionized a multitude of fields in the recent years and has provided state-of-the-art performances in fields such as computer vision and natural language processing [7], [8]. Due to its deep structure, DNN algorithms have the capability to learn complex patterns in data with multiple layers of abstraction [9], making them ideal candidates to learn complex patterns that exist in network traffic data. As a result, DNN based IDS (DNN-IDS) algorithms have received increased attention in recent work. A range of DNN algorithms, including but not limited to Feed Forward Neural Networks [10], Autoencoders [11], [12], [4], probabilistic models such as Restricted Boltzmann Machines (RBM) and Deep belief Networks (DBNs) [13], [14], [15] and recurrent neural networks such as LSTMs [16] have been used successfully in DNN-IDS. For a survey on DNN-IDS methodologies, readers are referred to [17].

Despite the impressive accuracies displayed by DNNs, DNNs are still used as black boxes. This is a major drawback in practical applications with human involvement as it provides little to no information about the reasoning behind a DNN prediction. This is especially true in DNN-IDS since most of the time, a human expert makes a higher level decision based on the recommendations of the automated system. Therefore, the user's trust on the DNN-IDS is imperative and providing justifications of the DNN-IDS predictions is as important as the prediction itself. In order to overcome these limitations, explaining DNNs has received increased attention in the recent years but still remains to be an open research area. David Gunning defines an explainable artificial intelligence system as one that answers the following questions. 1) "Why did it do that?" 2) "Why didn't it do something else?" 3) "When does it succeed?" 4) "When does it fail?" 5) "When can it be trusted?" and 6) "how can an error be corrected?" [18]. DNN explainability is categories into two broad categories,: 1) model transparency and 2) model functionality [19], [20]. Understanding what the network has learned and the reasons behind the concepts it has learned is model transparency [20]. Model functionality explanations, also known as post-hoc explanations is used to explain predictions by the model [19]. The work of this paper is focused on generating post-hoc explanations.

This paper presents a methodology for providing feedback to the user about the predictions of the DNN-IDS before and during deployment. The presented methodology provides feedback in the form of input feature contributions to the DNN-IDS predictions, i.e. the presented method presents the user with the input features which drove the prediction of the DNN-IDS toward an intrusion type. After training the DNN-IDS, prior to

deployment, the user is provided with the input features that are highly relevant to each intrusion type. During deployment, the user is able to see the driving factors of each individual classification decision. We argue that this insight improves transparency of the DNN-IDS and helps the user better understand the rationale behind DNN-IDS decisions Further, these feedback enables a user to qualitatively evaluate the DNN-IDS. The factors help build users' trust on the IDS.

The rest of the paper is organized as follows; Section II presents the methodologies for DNN-IDS and user feedback generation; Section III presents the experimental setup, results and a discussion and finally Section IV concludes the paper.

II. DNN-IDS and User Feedback Generation Methods

This section presents the Deep Neural Networks based Intrusion detection and the methodology for providing operator feedback.

A. DNN-IDS

In this paper, the IDS implemented as a feed forward neural network. The network is trained using supervised learning.

A cyber connection is considered as an input record ($X \in \mathbb{R}^d$). Each input record X is composed of a set of input features: $X = \{x_d\}$, where d denotes the dth feature in the dataset. Since supervised learning is employed, each X is associated with its label Y. In the DNN-IDS, Y indicates whether the cyber connection belongs to an attack group (abnormal) or normal communication. This can be binomial where the data it indicates the presence or absence of an attack, or it can be multinomial where the input record can be of a specific attack group. Therefore, the DNN-IDS learns a classification function, where the mapping can be expressed as $f: \mathbb{R}^d \rightarrow \mathbb{R}^+$.

The DNN-IDS consists of an input layer, an output layer and a multiple hidden layers. Each hidden neuron is activated as follows:

$$a_j^{(l+1)} = g\left(\sum_i a_i^{(l)} w_{ij}^{(l,l+1)} + b_j^{(l+1)}\right) \quad (1)$$

where, $a^{(l)}$ is the activation of the lth layer, $w_{ij}^{(l,l+1)}$ is the weight of the connection between ith neuron in layer l and jth neuron in layer $l+1$ and $b_j^{(l+1)}$ is the bias of the jth neuron in layer $l+1$.

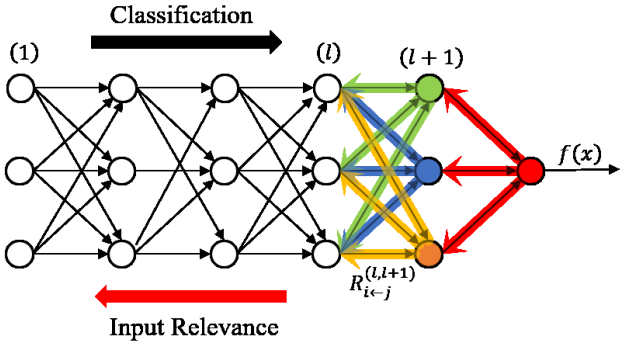

Fig 1: Back-propagation of relevance scores through the layers of the DNN using LRP method

Rectified Linear Unit (ReLu) neurons are used in the hidden layers, i.e. $g(\cdot)$ can be expressed as:

$$g(x) = \max(0, x) \quad (2)$$

The output layer is a softmax layer to obtain the probability distribution across the classes. Therefore, for each class the probability is calculated using the softmax function as follows:

$$P(Y = y_i|X) = \frac{e^{a_i}}{\sum_j e^{a_j}} \quad (3)$$

where the input values (a_i) are calculated using eq.(1) with $g(\cdot)$ being a linear pass through function.

After the softmax normalization, the cross entropy loss is minimized in the training of the DNN-IDS. The optimization process is carried out with a gradient based optimizer together with error back propagation. Dropout method is used for regularization of the DNN-IDS.

B. Calculating Input Feature Contributions

In this paper, the user feedback for the DNN-IDS is provided in terms of input feature relevance scores that indicate the contribution of each input feature to the detection of the intrusion. This is quantitative measure as to how much influence an input feature had in the prediction, which helps the user understand what input features drove the prediction of the DNN-IDS. The input feature relevance is calculated by decomposing the composite function of the DNN using a method named Layer-wise Relevance Propagation (LRP). It has to be noted that this section assumes that the DNN-IDS takes the form described in the subsection above.

Layer-wise relevance propagation (LRP) was introduced by Bach et al. as an approach for understanding the contribution of each pixel to image classification decisions made by the DNN [21], [22]. LRP assumes that the classification algorithm can be decomposed into several layers of computation. Inputs and the classification probabilities in the output layer are considered the first layer and the last layer, respectively. Each dimension of each layer has a relevance score ($R_d^{(l)}$) where d is the dimension and the l is the layer. The goal is to find the relevance scores for the layer l when the relevance scores for layer $(l+1)$ is available. Fig 1 depicts the high-level process of LRP.

The multilayered architecture of the DNN is leveraged by propagating the relevance scores in a backward pass, i.e. the relevance scores of a lower level is expressed as a function of upper level relevance scores. Relevance scores are back-propagated in "messages", $R_{i\leftarrow j}^{(l,l+1)}$ (from neuron j in $l+1$ to neuron i in l) such that a relevance conservation property holds as follows.

$$\sum_i R_{i\leftarrow j}^{(l,l+1)} = R_j^{(l+1)} \quad (5)$$

Similarly, the relevance score for ith neuron in the l layer can be expressed as:

$$R_i^{(l)} = \sum_j R_{i\leftarrow j}^{(l,l+1)} \quad (6)$$

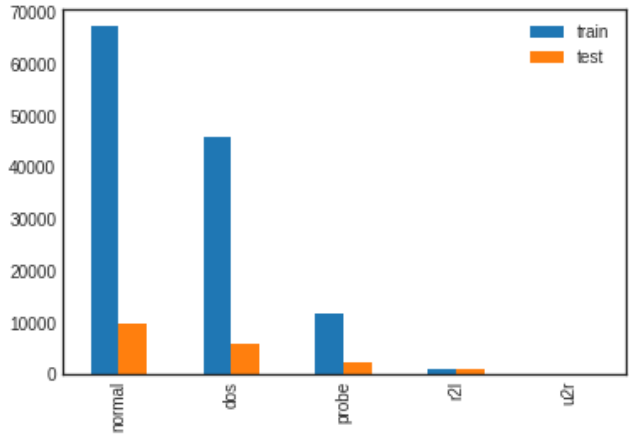

Fig 2: Data Distribution of the NSL-KDD Dataset

This relevance scores are distributed based on the ratio of pre-activations as follows:

$$R_{i\leftarrow j}^{(l,l+1)} = \left(\frac{a_i^{(l)} w_{ij}^{(l,l+1)}}{\sum_i a_i^{(l)} w_{ij}^{(l,l+1)} + b_j^{(l+1)}} \right) \cdot R_j^{(l+1)} \tag{7}$$

The drawback of the above propagation rule is that relevance scores become unboundedly large if the activations of the neuron (the denominator) is small. To alleviate this problem, the αβ method [21] is used as follows:

$$R_{i\leftarrow j}^{(l,l+1)} = \left(\alpha \frac{a_i w_{ij}^{+}}{\sum_i a_i w_{ij}^{+} + b_j^{+}} + \beta \frac{a_i w_{ij}^{-}}{\sum_i a_i w_{ij}^{-} + b_j^{-}} \right) \cdot R_j^{(l+1)} \tag{8}$$

Where, $a_i w_{ij}^{+}$ and b_j^{+} are the positive portion of the activations and the negative portion is indicated by a superscripted "-". Please note that the superscripted layer notations have been stripped off to simplify the notation and still i and j are considered to be indices associated with layers l and $l+1$ respectively. It has to be noted that the relevance decomposition satisfies following rule of relevance conservation:

$$f(x) = \cdots = \sum_{d\in l+1} R_d^{(l+1)} = \sum_{d\in l} R_d^{(l)} = \cdots = \sum_d R_d^{(1)} \tag{4}$$

Where $f(x)$ is the output and $R_d^{(1)}$ is the relevance score of the d^{th} dimension of the input layer.

When the relevance scores are propagated using the above method across layers to the input layer, the relevance score of each input feature d; $R_d^{(1)}$ can be obtained. It has to be noted that this calculation is done for each individual classification decision of the DNN-IDS. Further, it has to be noted that the un-normalized probabilities of the output layer are used for propagating the relevance.

C. Providing offline and online feedback to the user

As mentioned, the relevance scores of each input is calculated for each classification decision of the DNN-IDS.

In this work, offline feedback refers to providing user with information about what the DNN-IDS has learnt. This process takes place after training and prior to deployment of the DNN-IDS. The goal of providing this feedback is to enable user to qualitatively evaluate the DNN-IDS in addition to the quantitative evaluations done through accuracy scores. This analysis helps user see the concept the DNN-IDS has learnt about intrusions in general and about each intrusion type. To generate the offline feedback, a test dataset is. First, the test dataset is processed through the DNN-IDS to generate classification decisions. As mentioned in the above section, the un-normalized probability scores are used as the output from the DNN-IDS to generate the relevance scores. For offline feedback, mean relevance of each input feature for each intrusion type is reported to the user. With this information, the user can determine which factors drive the detection a certain intrusion type or intrusions in general. The mean relevance of each input feature per attack type is calculated as follows:

$$R_c^d = \frac{1}{N_c} \sum_k^{N_c} R_{x_{k_d}}^{(1)} \tag{10}$$

In this work, online feedback refers to providing the user with information about each individual classification that happens during the intrusion detection of a live data stream. In this step, the offline validated DNN-IDS is deployed on the network and is actively classifying the data stream. Therefore, in online feedback generation, LRP generated input relevance scores are presented for individual classification decisions. Therefore, the user is presented with the prediction from the DNN-IDS and the input features that drove the DNN-IDS to the prediction. If an attack is detected, without just giving the user the prediction, this system reports the inputs that indicate the detection of the attack.

III. Experiments

This section elaborates the experimentation procedure. First, the dataset and the preprocessing details are presented. Then, the two classification approaches – binary classification and multi-class classification – are presented. Finally a discussion on the results and the implications of the presented work is presented.

A. NSL-KDD Dataset and Preprocessing

The NSL-KDD dataset is an improved version of the KDD Cup '99 dataset [23]. The data were captured during the DARPA IDS evaluation program 1998 and was used in the Third International Knowledge Discovery and Data Mining Tools Competition. Tavallaee et al. created the NSL-KDD dataset by making improvements such as removing redundant records [23].

Each data record in the dataset is a TCP connection and they fall under five classes: normal, denial-of-service (DoS), probing, user-to-root (U2R), and root-to-local (R2L). The data distribution in the train and test datasets can be seen in Fig 2. It can be observed that the U2R and R2L classes are extremely underrepresented in the dataset creating a data-imbalance problem. Since solving a data imbalance problem is out of the scope of this work, the data records from R2L and U2R classes are not considered in this study.

TABLE I: TEST ACCURACY RESULTS – BINARY CLASSIFICATION

Model (hidden layers)	Classification Accuracy (%)	Accuracy Individual Class (%)	
		Intrusion	Normal
One (100)	96.19	95.04	97.16
Two (100, 50)	97.33	97.12	97.52
Three (100, 50, 80)	96.73	95.80	97.52

TABLE II: TEST ACCURACY RESULTS – MULTICLASS CLASSIFICATION

Model (hidden layers)	Classification Accuracy (%)	Accuracy Individual Class (%)		
		DoS	Probe	Normal
One (50)	93.83	97.75	68.03	97.74
Two (50, 80)	94.09	98.41	69.22	97.72
Three (30, 80, 30)	94.83	98.76	74.34	97.62

Each TCP connection record consist of 41 input features. These features consist of basic features acquired from the TCP connection, traffic features acquired from a window of two seconds and content featured acquired from the application layer data. In terms of data types, the 41 features contain 7 categorical features (four of them binary) and 34 continuous features. The categorical features pose a challenge in algorithms such as DNNs. In this work, the binary features are unchanged. The other three categorical features are converted to numerical features by using a simple label encoding scheme. Label encoding is used over one-hot encoding to preserve the number of input features and to avoid making the data sparse. In order to make sure that the features are considered in the same ranges, the input features were standardized to a zero mean and unit variance distribution.

B. Binary Classification Results

As mentioned, from the dataset, only the records belonging to the classes normal, DoS and Probe were used in this paper. For the binary classification, the focus was to train a DNN-IDS to identify between normal and abnormal (intrusion) traffic. Therefore, the both DoS and Probe were relabeled as intrusions and a binary classification was performed between normal and intrusion.

Different DNN-IDS models were built with different MLP architectures. This was performed to mainly to observe whether the factors that drive the intrusion detection changes with the depth of the DNN. Table I shows accuracy levels that was achieved by the different DNN-IDS models. Three MLP types in terms of depth was tested. The presented results are the best classification accuracies achieved for the particular depth. It has to be noted that very deep architectures were not tested in this study. All the models were able to achieve an overall test accuracy level of ~97% and it was observed that intrusions were detected at an accuracy of 97%.

LRP was implemented on the DNN-IDS to find the contributing factors for detecting intrusions. It was observed that the models allocated different importance levels to input features depending on the model. Fig 3 shows the average input feature relevance scores for data records classified as an intrusion. It can be observed that the DNN-IDS with a single layer paid the highest emphasis to "*percentage of connection that were rejected that was aimed at the same destination in the last two seconds*" (*rerror_rate*) and the "*percentage of connections to the same service out of the ones that were aimed at the same destination in the last two seconds*" (*same_srv_rate*), while "*percentage of connections that was rejected with the same destination IP*" (*dst_host_rerror_rate*) received highest emphasis with two hidden layers and *dst_host_rerror_rate* and *same_srv_rate* received highest emphasis for the DNN-IDS with three hidden layers.

C. Multi-class Classification Results

In this experiment, the DNN-IDS is trained to distinguish among three classes; normal, DoS and Probe. Similar to the two-class test, different models in terms of depth was implemented. Table II presents the test accuracies achieved for the different models. It was noticed that the models achieved similar overall test accuracies (~94%). Similarly, the DoS attack detection accuracy was similar across models (~98%). It was noticed that the probe attack detection suffered in accuracy. The highest probe attack detection accuracy was noticed in the model with three hidden layers (~75%).

Fig 4 shows the input feature relevance scores seen for detection of DoS attacks. As with binary classification, it was noticed that different input features received the highest relevance in driving the decision of the different DNN-IDS toward DoS. The model with single and three hidden layers showed similar behavior with the *same_srv_rate* and *dst_host_rerror_rate* receiving the highest relevance in DoS. The model with two hidden layers showed the highest relevance to *dst_host_rerror_rate*. The other input features that were considered by the models with high relevance was *rerror_rate*, *count*, and "*protocol type used in the connection*" (*protocol_type*).

Fig 4 shows the input feature relevance scores in Probe detection. It can be noticed that *rerror_rate* received the highest emphasis with the single hidden layer MLP. Whereas, models with two and three hidden layers gave the highest relevance to "*percentage of connections to different services with the same destination IP*" (*dst_host_diff_srv_rate*). Further, it was noticed that all models gave some relevance to *protocol_type,* "*whether user is logged in*" (logged_in), "*percentage of connections rejected with the same destination IP*" (*dst_host_srv_rerror_rate*), "*percentage of connections to different destinations with the same port as the current connection port in the last two seconds*" (*srv_diff_host_rate*), in detecting Probe.

D. Discussion

This generated feedback enables the users and domain experts to qualitatively validate the DNN-IDS before and during deployment. This gives the user the advantage of "peeking into" the black box of the DNN-IDS. The user gets to see the factors that cause the detection of an intrusion. A domain expert can assess these factors and determine whether the DNN-IDS is making decisions for the right reasons. This enables the users and the other domain experts to qualitatively evaluate the DNN-IDS in addition to the quantitative evaluation through accuracy scores.

In the experiments, it was noticed that despite similar accuracies, the different DNN-IDS made the decisions based on different features. This evidences the importance of having the

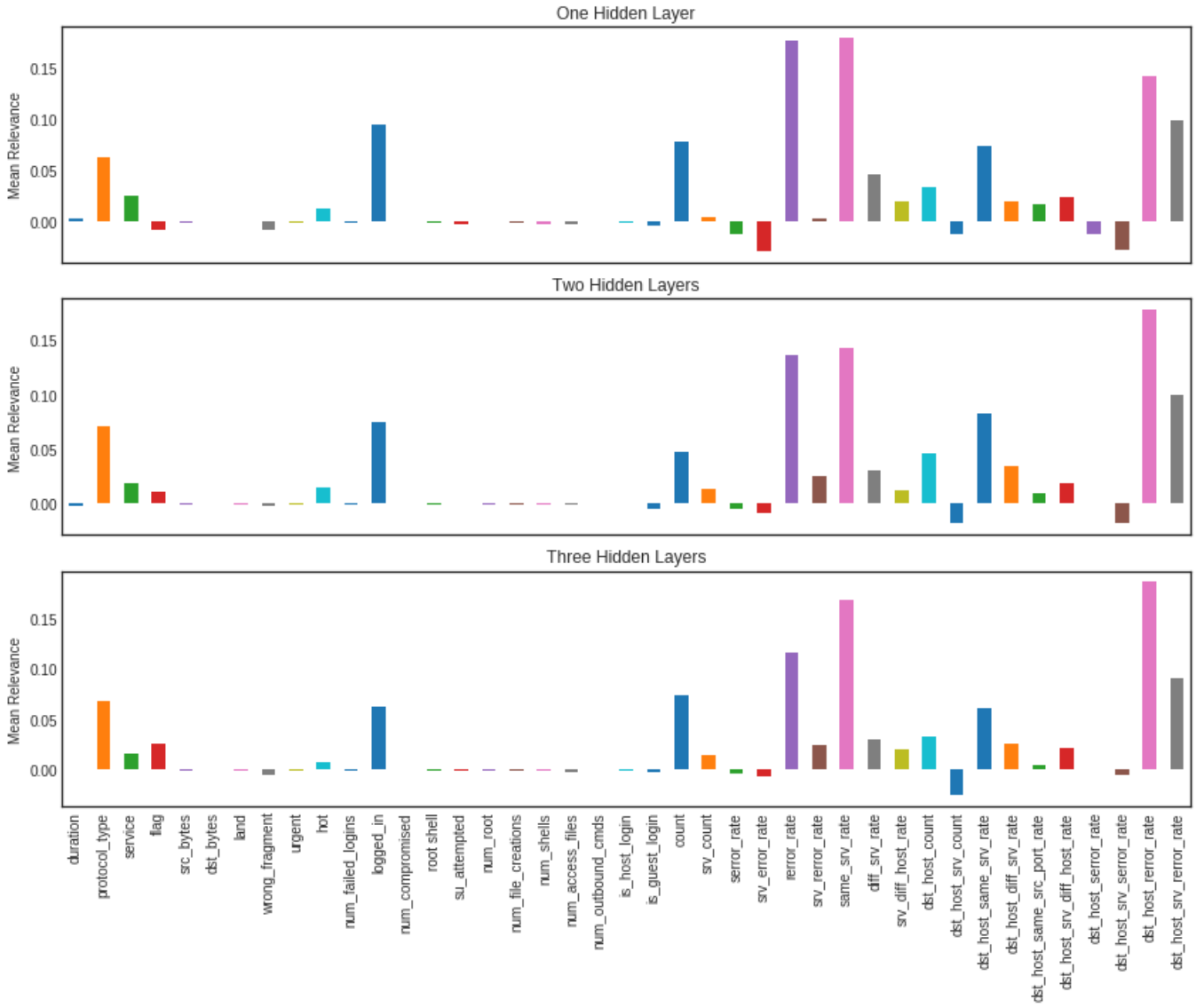

Figure 3: Input Feature relevance scores for class "Intrusion" in binary classification. (a) One hidden layer, (b) Two hidden layers (c) Three hidden layers. It can be noticed that although very similar, the priorities given to certain features change across models. The x-axis shows the abbreviated input feature names. The highest relevant features are explained in the text.

qualitative analysis of the DNN-IDS. The qualitative analysis can enable users to pick a model which makes decisions based on the reasons that align with domain expert knowledge. The combination of the qualitative and quantitative analysis leads to a better understanding of the DNN-IDS and its decision making process. Thus, it enables experts to use their experience to evaluate and analyze the DNN-IDS. This leads to building users' trust in the system.

IV. Conclusions and Future Work

This paper presented a methodology for improving the users' trust on DNN-IDS through generating user feedback on the decision making process of the DNN-IDS. The goal was to increase the transparency of the DNN-IDS which is necessary to build user trust. In this work, the feedback was presented by calculating the most relevant input features for predictions made by the DNN-IDS. A supervised learning based DNN-IDS (based on MLP) was considered. Two forms of feedback were generated: 1) offline feedback (after training, prior to deployment) and 2) online feedback (during deployment). In offline feedback, the user was provided the most relevant input features for each concept the DNN-IDS learned. This information enables the user to evaluate whether the input features that drive the decision of the DNN-IDS toward a certain class (e.g. attack type) aligns with the domain experts' knowledge. In online feedback, most relevant input features for each individual prediction is provided to the user. Then, the user can evaluate whether the prediction and the features driving the prediction align using their domain knowledge. This system was implemented on the KDD-NSL dataset for two intrusion types (DoS and Probe). Several DNN-IDS architectures with different depth were tested. It was observed that despite producing very similar accuracies, different architectures gave prominence to different features when identifying intrusions. Therefore, it can be concluded that the generated feedback can assist the user add

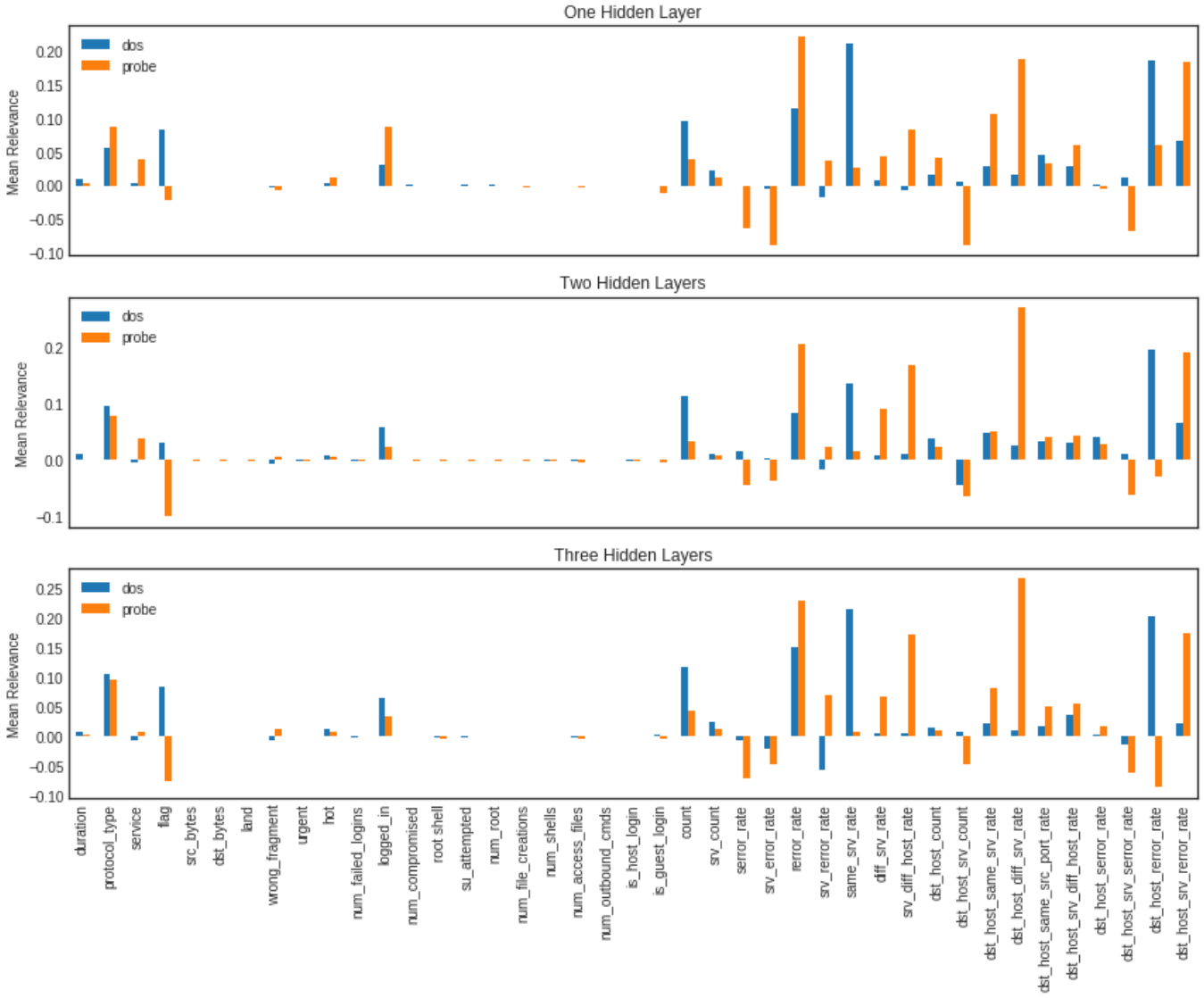

Figure 4: Input Feature relevance scores for "DoS" and "Probe" detection in three class classification. (a) One hidden layer, (b) Two hidden layers (c) Three hidden layers. It can be noticed that although very similar, the priorities given to certain features change across models. Input feature abbreviations are given in the x-axis

another layer of evaluation (qualitative) in addition to the quantitative evaluation through accuracy scores. The work needs to improve on the following fronts: 1) generating effective online feedback taking the speed of events into account, 2) generating feedback in easily understandable ways (e.g. textual descriptions).

REFERENCES

[1] R. Baheti and H. Gill, "Cyber-physical Systems."

[2] W. Wolf, "Cyber-physical Systems," *Computer (Long. Beach. Calif).*, vol. 42, no. 3, pp. 88–89, Mar. 2009.

[3] M. D. Valdes Pena, J. J. Rodriguez-Andina, and M. Manic, "The Internet of Things: The Role of Reconfigurable Platforms," *IEEE Ind. Electron. Mag.*, vol. 11, no. 3, pp. 6–19, Sep. 2017.

[4] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 2, no. 1, pp. 41–50, 2018.

[5] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," *Network, IEEE*, vol. 8, no. 3, pp. 26–41, 1994.

[6] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, 2016, pp. 581–585.

[7] S. Wang and J. Jiang, "Learning Natural Language Inference with LSTM," 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." pp. 1097–1105, 2012.

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

[10] O. Linda, T. Vollmer, and M. Manic, "Neural Network based Intrusion Detection System for critical infrastructures," in *2009 International Joint Conference on Neural Networks*, 2009, pp. 1827–1834.

[11] B. Zhang, Y. Yu, and J. Li, "Network Intrusion Detection Based on Stacked Sparse Autoencoder and Binary Tree Ensemble Method," *2018 IEEE Int. Conf. Commun. Work. (ICC Work.*, no. 61702046, pp. 1–6, 2018.

[12] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, “A Deep Learning Approach for Network Intrusion Detection System,” *Proc. 9th EAI Int. Conf. Bio-inspired Inf. Commun. Technol. (formerly BIONETICS)*, 2016.

[13] M. Z. Alom and T. M. Taha, “Network intrusion detection for cyber security using Deep Learning Approaches,” *Proc. Int. Jt. Conf. Neural Networks*, vol. 2017–May, pp. 3830–3837, 2017.

[14] M. Z. Alom, V. Bontupalli, and T. M. Taha, “Intrusion detection using deep belief networks,” *2015 Natl. Aerosp. Electron. Conf.*, pp. 339–344, 2015.

[15] M.-J. Kang and J.-W. Kang, “Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security.”

[16] Y. Chuan-long, Z. Yue-fei, F. Jin-long, and H. Xin-zheng, “A Deep Learning Approach for Intrusion Detection using Recurrent Neural Networks,” *IEEE Access*, vol. 5, pp. 1–1, 2017.

[17] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, “A survey of deep learning-based network anomaly detection,” *Cluster Comput.*, pp. 1–13, Sep. 2017.

[18] D. Gunning, “Explainable Artificial Intelligence (XAI) Explainable AI – What Are We Trying To Do ?,” *Def. Adv. Res. Proj. Agency*, pp. 1–18, 2017.

[19] Z. C. Lipton, “The Mythos of Model Interpretability,” Jun. 2016.

[20] S. Chakraborty *et al.*, “Interpretability of Deep Learning Models: A Survey of Results,” *IEEE Smart World Congr. DAIS - Work. Distrib. Anal. Infrastruct. Algorithms Multi-Organization Fed.*, 2017.

[21] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PLoS One*, vol. 10, no. 7, pp. 1–46, 2015.

[22] A. Binder, G. Montavon, S. Lapuschkin, K. R. Mï¿½ller, and W. Samek, “Layer-wise relevance propagation for neural networks with local renormalization layers,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9887 LNCS, pp. 63–71, 2016.

[23] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” *IEEE Symp. Comput. Intell. Secur. Def. Appl. CISDA 2009*, no. Cisda, pp. 1–6, 2009.