Deep Learning-Based Detection of Cyberattacks in Software-Defined Networks

Seyed Mohammad Hadi Mirsadeghi[1(✉)], Hayretdin Bahsi[1], and Wissem Inbouli[2]

[1] Department of Software Science, Tallinn University of Technology, Tallinn, Estonia
semirs@teltech.ee, hayretdin.bahsi@taltech.ee
[2] University of Lorraine, LORIA, Nancy, France
wissem.inbouli@loria.fr

Abstract. This paper presents deep learning models for binary and multiclass intrusion classification problems in Software-defined-networks (SDN). The induced models are evaluated by the state-of-the-art dataset, InSDN. We applied Convolutional Autoencoder (CNN-AE) for high-level feature extraction, and Multi-Layer Perceptron (MLP) for classification that delivers high-performance metrics of F1-score, accuracy and recall compared to similar studies. Highly imbalanced datasets such as InSDN underperform in detecting the instances belonging to the minority class. We use Synthetic Minority Oversampling Technique (SMOTE) to address dataset imbalance and observe a significant detection enhancement in the detection of minority classes.

Keywords: Software-Defined Network · Intrusion Detection · Deep Learning · Dataset Balancing

1 Introduction

Software-Defined Networking (SDN) has revolutionized network management through the physical separation of the control plane from forwarding devices. Through the decoupling of control and data planes in the SDN paradigm, all network intelligence and control logic is migrated from network devices to a logically centralized software-based entity known as the network controller. Thus, SDN provides network-wide visibility, centralized network intelligence, and network programmability. However, these features and the SDN architecture itself introduce new security risks and attack vectors that are not present in conventional network deployments. The new attack surface introduced by SDN is due to the inherent alterations to network components and the relationships between them. For example, centralized architecture brings about a single point of failure (SPOF). In other words, if the network controller is compromised by an adversary, the entire network may be in jeopardy. Moreover, SDN elements themselves may be used as reflectors in Amplification DDoS attacks.

S. Goel et al. (Eds.): ICDF2C 2022, LNICST 508, pp. 341–354, 2023.
https://doi.org/10.1007/978-3-031-36574-4_20

Intrusion detection is a critical detective security function for identifying cyber attacks usually in near real-time and initiating the relevant course of action. The common solutions depend on attack signatures, meaning that such intrusion detection systems (IDSs) have a limitation to detect future variations or enhancements of the existing attack types. Machine learning-based approaches promise to detect new attack types, eliminating the need for signature generation tasks. Instead, they learn from the data. In an SDN architecture, security functions such as intrusion detection can be embedded as a software application on top of the controller or it can be deployed as an independent system component. Thus, a machine learning-based detection function can be applied in various ways.

In this paper, we introduce deep learning-based models for detecting intrusions in software-defined networks. More specifically, we have developed two binary classification models, one of them focusing on DDoS detection and the other one discriminating all intrusions from normal traffic. We also created a detection system for identifying intrusion types, which is achieved by a model that performs multi-class classification. We observed very low detection rates for minority classes. Thus, we applied balancing strategy to boost the performance. We used a relatively new dataset [8] that is generated in an SDN network. There exist some studies (e.g., [2]) that apply deep learning models without a dataset balancing strategy. This study demonstrates that balancing is a significant enabler for better performance.

The structure of this paper is as follows: Sect. 2 gives background information about the subject and reviews the literature. The machine learning workflow and the applied methods are introduced in Sect. 3. The results are given and discussed in Sect. 4. Section 5 concludes the paper.

2 Related Work

The industry is exploring ways to exchange information among trusted parties to improve network security. A collaborative concept called DDoS eXchange Points (DXP) has been proposed [20] based on combining information from multiple sources across the network. Recent studies [14] suggest combining ML and SDN to construct a scalable and accurate bot-detection framework. Such a framework leverages centralized learning with distributed detection to achieve better accuracy. In 2016, a bot detection scheme in SDNs was proposed [16] by gathering centralized network flow statistics and applying a decision tree that depends on a supervised machine learning classification algorithm which achieved an 80% detection rate over a publicly available real-world botnet dataset. In 2017, a flow-based approach to detecting botnets in software-defined networks was proposed [17] using ML algorithms without the need for reading packet payload which achieved a 90% accuracy testing their method on publicly available CTU-13 botnet datasets and ISOT botnet datasets. In 2018, an SDN framework for detection of defense against DDoS attacks was proposed [21]. This framework consists of a traffic collection module, flow table delivery module as well as DDoS attack detection module that was built using the SVM model. This experiment

was conducted using the KDD99 dataset demonstrating a high accuracy rate of 99.8%. Even though the KDD99 dataset has been widely used for testing and training network intrusion detection datasets, it does not resonate with Software-defined networks and may pose compatibility issues.

Deep Learning-based approaches have outperformed existing machine learning techniques when applied to various classification problems in SDN networks [11]. CNN-based models for Intrusion detection in Software-defined Networks have been presented [7]. Hybrid CNN-LSTM Model for anomaly detection has been presented [1] by training a model on the InSDN dataset both in space and time. Moreover, a Gated Recurrent Unit Recurrent Neural Network (GRU-RNN) was proposed for intrusion detection systems in SDN [15] which was only tested on classical intrusion dataset NSL-KDD. It is apparant that the research community can benefit from more intrinsic SDN data.

3 Method

3.1 Dataset

Even though SDN is increasingly deployed in cloud computing infrastructure and globally deployed in WANs, the technology is still under development. While classical intrusion detection datasets such as KDD'99 [6] and NSL-KDD [18] provide us with cases of various attacks vs normal traffic, they do not reflect the characteristics of SDN. Actual traffic data captured from data centers and globally deployed WANs such as Google B4 [9] and Espresso [22] can shed a light on the specifics of DDoS detection in SDN but they are not publicly available. Even though helpful in understanding how to detect attacks in traditional networks, using a non-specific SDN dataset may cause compatibility problems as attack vectors would not resonate with the SDN architecture.

For our study, we leverage InSDN [8], a recent specific attack dataset obtained from an SDN environment. InSDN dataset contains a total number of 343,939 network flow traces where normal traffic brings 68,424 instances and attack traffic is split into seven different attack classes. DDoS and Probe attacks are represented by 121,942 and 98,129 instances, respectively [8]. The remaining one per cent of attack traffic consists of 53616 DoS, 1405 BFA, 192 web attack, 164 botnet and only 17 U2R instances, indicating an outstanding imbalance in the dataset. It is common to describe the imbalance of classes in terms of a ratio of a class with respect to the largest class. For instance, the imbalance ratio of U2R compared to DDoS is 1:7173.

This dataset provides flow-level features such as flow duration, inter-arrival time, number of packets, and number of bytes. The dataset also includes statistical features such as the maximum, minimum, mean, or standard deviation of the flow-level features. These features can be directly extracted from the SDN controller through API queries or by extracting from the flow data.

3.2 Overview of the Classification Tasks

Deep learning models learn the data representation with multiple levels of abstraction. These methods have dramatically improved the model performances

in speech recognition or visual object recognition among others [3]. In this study, we seek to design, implement and evaluate deep learning architectures that learn the structures of the dataset for three different supervised classification problems: (1) A binary classifier that discriminates DDoS instances from the normal ones (DDoS classification), (2) A binary classifier that recognizes whether an instance is an intrusion (i.e., not limited to DDoS) or not (Intrusion classification), (3) Multi-class classifier that identifies the exact type of intrusion (Intrusion Type classification).

The experiments were carried out using Python and the Pytorch [12] library. The computing resource has a processor with 2199.998 MHz, 13G memory and 56320K L3 cache.

3.3 Binary Classification

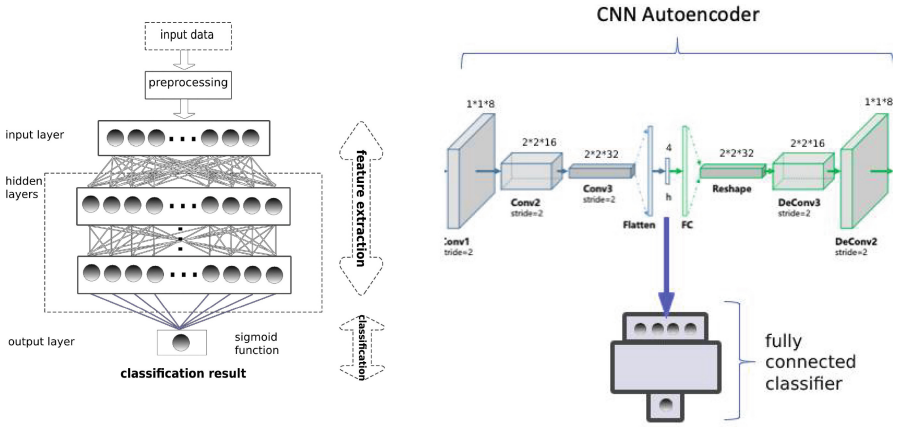
We induced three separate deep-learning models for DDoS and intrusion classification problems:

(1) A deep learning architecture composed of various neural network layers for the raw features (Basic Architecture),
(2) A similar deep learning architecture but obtains high-level features from an autoencoder architecture (Autoencoder Architecture),
(3) A simple ensemble model that combines the decision of results obtained from (1) and (2) (Simple Ensemble Architecture).

For the basic architecture, we included a 7-layer neural network with 52, 128, 512, 512, 128, 64, and 16 nodes respectively. The rectified linear activation function (ReLU) has been used at each layer to increase the non-linearity degree and set all negative values in the feature set by zero. Finally, the output layer incorporates the Sigmoid function to represent the probability of each input flow belonging to either class. We use the Adam optimization algorithm for stochastic gradient descent where the learning rate is set to 0.0001 and train the deep learning model. At each backpropagation step during the training process, we calculate the reconstruction loss using Binary Cross Entropy criterion (Fig. 1).

Autoencoder, generally deployed as a generative model, is proficient at extracting high-level features from data by transforming it into a latent space by an architecture composed of an encoder, bottleneck and decoder. The latent space representation of data lies at the bottleneck layer of Autoencoder, which is later used to generate new data instances by mapping the original instances to the new space. Convolutional Autoencoders have proven to be excellent at denoising data. Thus, for the second model, we design a Convolutional Autoencoder and train it over the InSDN dataset. Then the bottleneck representation of the dataset is fed to a deep learning framework.

The architecture of the convolutional autoencoder is composed of three convolutional layers taking in flow-level values through one input channel and expanding it first to 8 and next to 16 and 32 channels. Each channel corresponds to a different filter applied throughout the convolution process. The output of



(a) Basic Architecture (b) Autoencoder Architecture

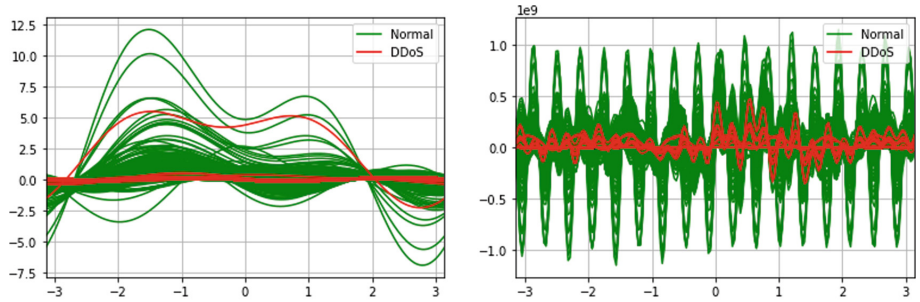
Fig. 1. Deep Learning models

convolutional layers is then flattened and linearly transformed into the bottleneck space dimension of [1,4]. The decoder has the same architecture as the encoder except in reverse order. Throughout the training process, the decoder reconstructs training data instances with the aim of minimizing the loss between the original and reconstructed ones.

As a first pre-processing step, we reshape each flow instance of dimensions [1,52] into an image-like structure of shape [8,8]. This transformation is carried out to suit the input dimensions of the convolutional autoencoder using a linear function (52,64) which then is unflattened to shape [8,8]. Convolutional autoencoder is trained over a train set for 200 epochs leaving us with a latent space representation view of network traffic data. In this fashion, we can extract 4 high-level features out of 52 features. As depicted in Andrew's curve for high-level features (Fig. 2a), we observe more linearity in feature space as the curves for normal and DDoS behavior are definitively less tangled and more easily separable for DDoS classification than the original features (Fig. 2b).

We insert the 4-dimensional output from the bottleneck of the Convolutional Autoencoder into a deep learning classifier with the architecture of a fully-connected classifier consisting of six linear layers going from 4 to 16, 32, 32, 16, and 1 node at each layer. We use rectified linear unit activation function at each layer.

The third one, the basic ensemble model, combines the classification results of the first and second models. Aggregation of information from the two models may result in decisions that are superior to those made by either single model. We combine the proposed classification models in such a fashion that they share their decisions on each flow instance. If either model decided that a flow belongs to an attack category, then it is considered an attack. In the case that both models agree on a flow instance being normal, then it is considered normal.



(a) High-level features (b) Raw features

Fig. 2. Comparison of Andrew's Curves for raw and high-level features

3.4 Multiclass Classification and Dataset Balancing

For the intrusion type classification problem, we propose a 10-layer fully connected neural network with 52, 128, 512, 1024, 1024, 512, 512, 128, 64, 16, and 8 nodes in each layer. The architecture takes 52 features, gradually expands across the data space, obtains high-level abstractions of training data and ultimately measures class probabilities at the output nodes.

The intrusion class of the InSDN dataset has 7 distinct intrusion types which are listed with their instance numbers in Table 1. The majority of traffic flows belong to DDoS, DoS and Probe attack classes with the remaining classes making up about 1% of the dataset (e.g., an imbalance ratio of 1:7173 between U2R and DDoS). Thus, the model may not effectively learn the decision boundary in the case of minority classes which are also important to identify.

We use Synthetic Minority Over-sampling Technique (SMOTE) to address the imbalance issue of InSDN Intrusion dataset. In this technique, minority classes are over-sampled by creating synthetic examples rather than by over-sampling with replacement. In this fashion, minority classes are over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending on the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen [5].

A dataset is imbalanced if its classes are not approximately equally represented. Imbalance on the order of 100 to 1 is prevalent in fraud detection and imbalance of up to 100,000 to 1 has been reported in other applications [13]. Deep learning models trained with imbalanced cybersecurity data cannot recognize minority classes effectively. One way to address this issue is to use resampling in an attempt to adjust the ratio between different classes to obtain a balanced dataset.

The research community has addressed the notion of class imbalance mainly in two ways. One approach is to assign distinct costs to training examples [10].

The other angle is to re-sample the original dataset, either by over-sampling the minority class and/or under-sampling the majority class [4].

Table 1. The number of instances in the dataset before and after data augmentation

Intrusion Type	Number of instances in Original Source	Number of instances after data augmentation
BFA	1405	50181
U2R	17	48793
Probe	98129	146905
DoS	53616	78004
DDoS	121942	121942
Web-Attack	192	48968
Botnet	164	48940
Normal	68424	79127

Authors of SMOTE [5] suggest combining the Synthetic Oversampling Technique with random undersampling of the majority class. Therefore, we intend to first oversample the minority class to have as many more examples as 25 per cent of the number of majority class, then use random undersampling to reduce the number of examples in the majority class by 75 per cent of the minority class. The balanced intrusion dataset includes 636545 samples in total. Table 1 reports dataset statistics before and after the data augmentation.

We conducted separate experiments by training our 10-layer fully connected neural network with original and balanced datasets for a total of 200 epochs. Both experiments were carried out using the same test set obtained before balancing.

3.5 Evaluation Criteria

To evaluate the performance of our classifiers, we use classification accuracy, precision, recall, and F1 score as performance metrics. In addition, we calculate the confusion matrix where:

- `True Positive (TP)` indicates attack traffic correctly classified as malicious(DDoS).
- `True Negative (TN)` indicates normal traffic correctly classified as benign.
- `False Positive (FP)` indicates normal traffic incorrectly classified as malicious.
- `False Negative (FN)` indicates attack traffic incorrectly classified as normal.

4 Results and Discussion

4.1 Data Pre-processing

Socket information such as Source IP, Destination IP, and flow ID is removed to avoid the over-fitting problem given that these features can be changed from network to network. The original dataset includes 77 features [8]. InSDN dataset also includes as many as 8 zero variation features that do not contain any information useful for classification purposes. These features (**Fwd Byts/b Avg', 'Fwd Pkts/b Avg', 'Fwd Blk Rate Avg', 'Bwd Byts/b Avg', etc.**) along with TCP flags are removed, leaving us with a total of 52 numerical features. Finally, features have different ranges, so they need to be standardized to restrict the scale of values.

We split more than 189000 network flow traces provided by the InSDN dataset (i.e., DDos and normal instances)for DDoS traffic classification into training and test sets with an 80% train, 20% test ratio.

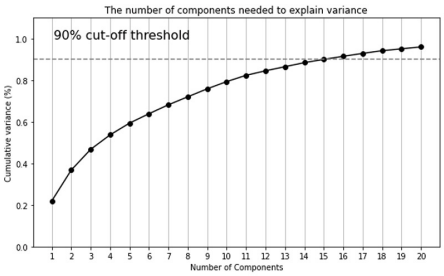
4.2 Explotary Data Analysis

Our feature set includes basic features such as the duration of a flow and flow Inter-arrival time as well as statistical features such as min, max, and standard deviation of the basic features. First, we use the Pearson correlation coefficient to compute the pair-wise correlation matrix for all features. We conducted this analysis for the intrusion classification problem. This matrix reveals that time-related features such as Flow Duration, Flow Inter-Arrival Time, and Idle Std seem to be correlated. Next, we reduce the dimensionality of the feature space by applying a principle component analysis (PCA). A PCA decomposition can be used to project a high-dimensional space to a lower-dimensional space by relying on the initial principal components. In effect, it converts a set of values of M possibly correlated variables into a set of K uncorrelated variables, the PCAs. We find that a significant number of the features are correlated since the first 8 PCAs explain more than 70% of the variance and the first 15 about 90% as shown in Fig. 3a. In 3b, it is observed that the first and second PCA vectors provide relatively distinguishable feature space projection for the intrusion classification task.

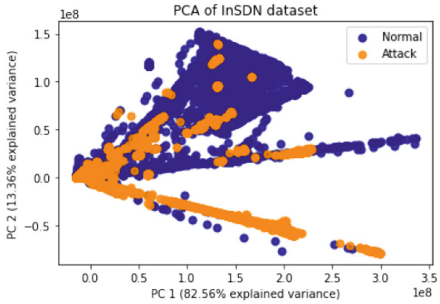
4.3 Experimental Results: Binary Classification

DDoS classifier with Basic Architecture is trained over the train set in the course of 200 epochs and demonstrates a remarkable accuracy of 99.72%.

In the context of DDoS detection, a false negative means an attack instance was missed. While our basic architecture performs remarkably for DDoS classification, there are 6 false negatives in the confusion matrix (Fig. 4a). We also observe 99 normal flow instances that were classified as DDoS instances by this architecture.



(a) Cumulative Explained Variance by Principal Components



(b) Benign vs. Malicious: projection of feature space to 2 Principle Components.

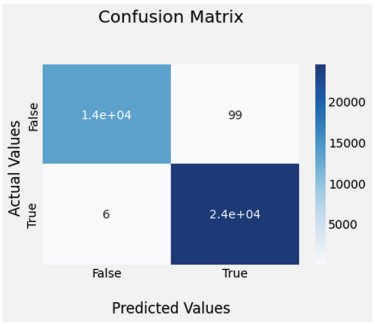
Fig. 3. PCA Results of Intrusion Classification

Table 2. Binary Classification Results

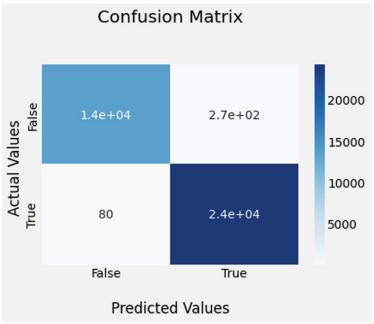
	DDoS Classification				Intrusion Classification			
Model	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Basic	**99.72**	**99.59**	99.97	99.78	99.67	99.64	99.94	99.79
Autoencoder	99.08	98.90	99.67	99.28	91.53	95.04	94.35	94.69
Simple Ensem.	99.14	98.70	**99.98**	99.33	**99.71**	**99.66**	**99.97**	**99.81**

Autoencoder architecture is trained with the same train dataset for a total of 4000 epochs. Although autoencoder enabled us to represent the whole feature space by only 4 features, the performance of this architecture has slightly less than the basic one in all metrics as shown in Table 2 for DDoS classification. The confusion matrix given in Fig. 4b depicts that false positives and false negatives increase in this model type. However, a simple ensemble model that merges the results of the other two models provided a better recall value, decreasing the false negative samples (see Fig. 4c). Still, the basic architecture is better in other metrics, accuracy, precision and F1-score due to the lower false positive number.

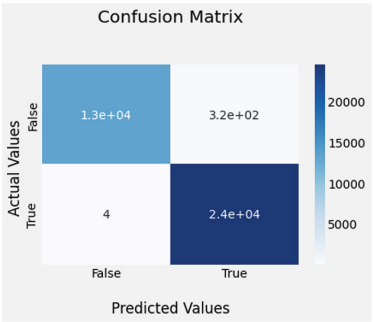
Our experiments regarding the intrusion classification indicate similar results to the case of DDoS classification except the simple ensemble model is slightly better than all other models in all metrics as shown in Table 2. Still, the autoencoder model has the worst performance when it is individually evaluated. However, this model contributes to the simple ensemble model positively and slightly boosts its performance when used with basic architecture in tandem. The number of false negative instances obtained from a simple ensemble decreases to 13 from 28 when compared to the basic architecture (see Fig. 5).



(a) Confusion Matrix of Basic Architecture



(b) Confusion Matrix of Autoencoder Architecture



(c) Confusion Matrix of Simple Ensemble Architecture

Fig. 4. Confusion Matrices for DDoS Classification

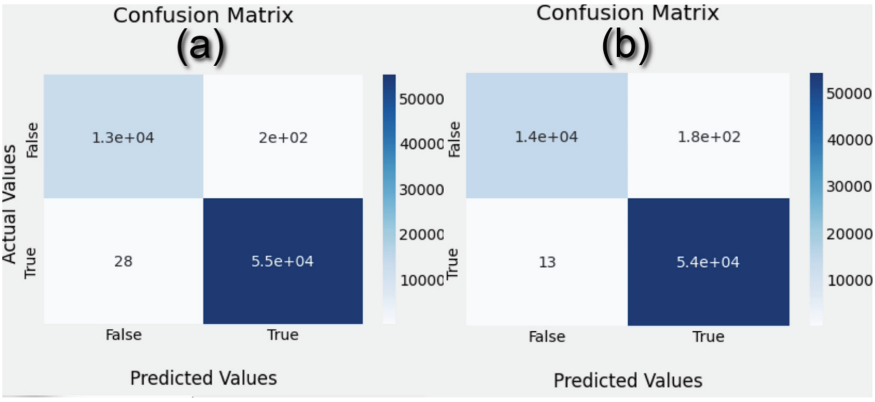


Fig. 5. Confusion Matrices of intrusion classification: (a) Basic Architecture (b) Simple Ensemble Architecture

4.4 Results of Intrusion Type Classification and Data Balancing

The unbalanced nature of a dataset may have implications on the multiclass classification setting as minority classes may not be identified by the model due to their limited representation in the training datasets. We observed this limitation in our experimental results regarding the intrusion type classification. As demonstrated in Table 3, despite very accurate results for the intrusion types, DDoS, DoS and Probe, the detection rates of BFA, Web-Attack and botnet are lower. In a more extreme case, any instance belonging to the intrusion type, U2R, has not been identified in the test dataset.

When the model is trained with a balanced dataset (i.e., recall that we use SMOTE as a balancing method in this study), we obtain a remarkable increase in detection rates of minority classes. As shown in Table 3, the new model is able to detect approximately 67% of the U2R attacks. We also observed significant increases in the detection rates of Botnet, Web-Attack and BFA.

Table 3. F1 Performance of Multi-class Classification over balanced dataset

Class	BFA	Botnet	DDoS	DoS	Normal	Proble	U2R	Web-Attack
balanced dataset	95.72	100	99.99	99.81	99.74	99.05	66.66	100
unbalanced dataset	82.43	57.57	99.98	99.90	99.92	99.79	0	44.73

The study, [2], proposes multiple deep-learning models for detecting Denial-of-Service attacks in the InSDN dataset. Their models induced by RNN, LSTM, and GRU models [2] proposed to take in 48 features from InSDN dataset and perform adequately by measures of precision, recall, and F1-scores. V-NKDE, an ensemble model that incorporates Voting -Naive Bayes, K Nearest Neighbors, Decision Tree, and Extra Trees has been proposed by [19]. The concept behind the voting classifier is to merge various machine learning classifiers conceptually and use a majority vote to predict class labels. Both of these studies provide multiclass classification models for the same dataset that we used in this study.

As demonstrated in Table 4, our model for the balanced dataset achieves the highest detection rates for BFA, Botnet, Web-Attack and U2R. Despite the fact that [2] uses relatively advanced models for grasping the temporal aspect of the dataset and [19] uses a well-developed ensemble method, our results for DDoS, DoS and Probe are very close to their performances and we obtained higher detection rates for minority classes. These results indicate that balancing the dataset in this problem domain would be essential for better performance regardless of the complexity of the utilized ML algorithm.

Table 4. Comparison of Multiclass Classification Results

Type Detection	per-class F1-score (%)						
Model	*BFA*	*Botnet*	*DDoS*	*DoS*	*Probe*	*U2R*	*Web-Attack*
RNN[2]	75.6	79.51	99.93	98.82	98.60	21.05	12.78
LSTM[2]	80.50	82.50	99.94	97.87	98.18	12.90	13.80
GRU[2]	80.33	53.65	99.94	98.11	98.48	36.36	12.84
V-NKDE[19]	92.7	95.7	100	99.9	99.9	not mentioned	69.3
Our Balanced Model	95.72	100	99.99	99.81	99.05	66.66	100

5 Conclusion

InSDN dataset paves the way for the research community to develop machine learning models for detecting intrusions in SDNs. In this study, we propose deep learning-based models for binary and multiclass classification problems by using this dataset. We developed two distinct binary models, one for discriminating DDoS from normal traffic and one for detecting any type of intrusion. The multiclass model identifies the type of intrusion among 8 different classes that also include the normal class.

We investigated whether low-dimensional vectors obtained from Convolutional Encoders improve the performance of a basic deep-learning model composed of various neural network layers. We found that utilizing such vectors for model building slightly reduces the number of false negatives but creates many false positives. A simple ensemble that combines the outputs of two models, one created with raw features and one with vectors of autoencoder layer slightly increases performance, indicating that still, weaker models enhance detection when they contribute to the ensemble rather than acting as a standalone model.

We applied a balancing strategy, SMOTE, in the context of multiclass classification and achieved better results in identifying the minority intrusion types which can create significant harm to their targets. When we compared our results with the findings of studies that address the same dataset, we deduced that the application of a balancing strategy would be more instrumental than the complexity of the machine learning algorithm.

Acknowledgement. This work is partially funded by the European Union's Horizon 2020 Research and Innovation Programme through ECHO (https://echonetwork.eu/) project under Grant Agreement No. 830943.

References

1. Abdallah, M., An Le Khac, N., Jahromi, H., Delia Jurcut, A.: A hybrid CNN-LSTM based approach for anomaly detection systems in SDNs. In: The 16th International Conference on Availability, Reliability and Security, pp. 1–7 (2021)
2. Alshra'a, A.S., Farhat, A., Seitz, J.: Deep learning algorithms for detecting denial of service attacks in software-defined networks. Procedia Comput. Sci. **191**, 254–263 (2021)

3. Bengio, Y., LeCun, Y., et al.: Scaling learning algorithms towards AI. Large-scale Kernel Mach. **34**(5), 1–41 (2007)
4. Chawla, N.V.: Data mining for imbalanced datasets: an overview. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, pp. 875–886. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-09823-4_45
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
6. Divekar, A., Parekh, M., Savla, V., Mishra, R., Shirole, M.: Benchmarking datasets for anomaly-based network intrusion detection: KDD CUP 99 alternatives. In: 2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS), pp. 1–8. IEEE (2018)
7. Elsayed, M.S., Jahromi, H.Z., Nazir, M.M., Jurcut, A.D.: The role of CNN for intrusion detection systems: an improved CNN learning approach for SDNs. In: Perakovic, D., Knapcikova, L. (eds.) FABULOUS 2021. LNICST, vol. 382, pp. 91–104. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-78459-1_7
8. Elsayed, M.S., Le-Khac, N.A., Jurcut, A.D.: InSDN: A novel SDN intrusion dataset. IEEE Access **8**, 165263–165284 (2020)
9. Jain, S., et al.: B4: experience with a globally-deployed software defined wan. ACM SIGCOMM Comput. Commun. Rev. **43**(4), 3–14 (2013)
10. Margineantu, D.: Building ensembles of classifiers for loss minimization. Comput. Sci. Stat., 190–194 (1999)
11. Niyaz, Q., Sun, W., Javaid, A.Y.: A deep learning based DDoS detection system in software-defined networking (SDN). arXiv preprint arXiv:1611.07400 (2016)
12. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32, pp. 8024–8035. Curran Associates, Inc. (2019). http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
13. Provost, F., Fawcett, T.: Robust classification for imprecise environments. Mach. Learn. **42**(3), 203–231 (2001)
14. Shinan, K., Alsubhi, K., Alzahrani, A., Ashraf, M.U.: Machine learning-based botnet detection in software-defined network: a systematic review. Symmetry **13**(5) (2021). https://doi.org/10.3390/sym13050866. https://www.mdpi.com/2073-8994/13/5/866
15. Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M.: Deep recurrent neural network for intrusion detection in SDN-based networks. In: 2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), pp. 202–206. IEEE (2018)
16. Tariq, F., Baig, S.: Botnet classification using centralized collection of network flow counters in software defined networks. Int. J. Comput. Sci. Inf. Secur. **14**(8), 1075 (2016)
17. Tariq, F., Baig, S.: Machine learning based botnet detection in software defined networks. Int. J. Secur. Appl **11**(11), 1–12 (2017)
18. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 1–6. IEEE (2009)
19. Tayfour, O.E., Marsono, M.N.: Collaborative detection and mitigation of DDoS in software-defined networks. J. Supercomput. **77**(11), 13166–13190 (2021)
20. Wagner, D., et al.: United we stand: collaborative detection and mitigation of amplification DDoS attacks at scale. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, pp. 970–987 (2021)

21. Yang, L., Zhao, H.: DDoS attack identification and defense using SDN based on machine learning method. In: 2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN), pp. 174–178. IEEE (2018)
22. Yap, K.K., et al.: Taking the edge off with espresso: scale, reliability and programmability for global internet peering. In: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 432–445 (2017)