A Few-shot Deep Learning Approach for Improved Intrusion Detection

Conference Paper · October 2017

DOI: 10.1109/UEMCON.2017.8249084

4 authors, including:

Md Moin Uddin Chowdhury
North Carolina State University
2 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Jiang Li
Old Dominion University
114 PUBLICATIONS **749** CITATIONS
SEE PROFILE

A Few-shot Deep Learning Approach for Improved Intrusion Detection

Md Moin Uddin Chowdhury*, Frederick Hammond†, Glenn Konowicz‡, Jiang Li§,Chunsheng Xin¶ and Hongyi Wu‖
Department of Electrical and Computer Engineering, Old Dominion University
Email: * mchow001@odu.edu, †dowufred13@yahoo.com, ‡gkono001@odu.edu, §jli@odu.edu,
¶cxin@odu.edu, ‖h1wu@odu.edu,

***Abstract*—Our generation has seen the boom and ubiquitous advent of Internet connectivity. Adversaries have been exploiting this omnipresent connectivity as an opportunity to launch cyber attacks. As a consequence, researchers around the globe devoted a big attention to data mining and machine learning with emphasis on improving the accuracy of intrusion detection system (IDS). In this paper, we present a few-shot deep learning approach for improved intrusion detection. We first trained a deep convolutional neural network (CNN) for intrusion detection. We then extracted outputs from different layers in the deep CNN and implemented a linear support vector machine (SVM) and 1-nearest neighbor (1-NN) classifier for few-shot intrusion detection. few-shot learning is a recently developed strategy to handle situation where training samples for a certain class are limited. We applied our proposed method to the two well-known datasets simulating intrusion in a military network: KDD 99 and NSL-KDD. These datasets are imbalanced, and some classes have much less training samples than others. Experimental results show that the proposed method achieved better performances than the state-of-the-art on those two datasets.**

***Index Terms*—Intrusion Detection System(IDS); low shot learning; CNN; SVM.**

I. Introduction

It is estimated that there will be roughly 50 billion devices that will connect to the Internet by year 2020. To keep abreast with this exponential pace of Internet growth, cyber attacks by hackers will exploit new flaws in Internet protocols, operating systems and application software. There exists several protective measures such as firewall which is placed at the gateway to check activities of intruders. To meet the dynamic characteristics of attacks, *intrusion detection systems* (IDSs) [1] is used as a second line of defense. IDSs dynamically monitor network log, file system, and real-time events occurring in a computer system or network and analyze them for signs of adversaries or attacks [2]. IDSs are classified as host-based or network-based. Host-based IDSs operate on information collected from within an individual computer system, while network-based IDSs collects raw network packets as the data source from the network and analyze for signs of intrusions [2]. There are two different detection techniques, misuse detection and anomaly detection, employed in IDSs to search for attack patterns. Misuse detection systems find known attack signatures in the monitored resources, whereas anomaly detection systems identify attacks by detecting changes in the pattern of utilization or behavior of the system. However, at present, anomaly detection IDSs have not been widely adopted. On the other hand, despite having limited potential against unfamiliar attacks, misuse detection systems find greater usage in commercial arena [3].

With the increasing processing power of modern CPUs, data mining/machine learning technique has become an alternative to manual human input. This approach was first introduced in mining audit data for dynamic and automatic models for intrusion detection (MADAMID) using association rules [4]. However, the majority of IDSs currently in use are prone to generation of false positive alarms. To this end, there are only few datasets reflecting actual network connections being publicly available for classifying normal from abnormal connections. Among them KDD 99 and NSL-KDD [3] are well known public datasets to promote anomaly detection techniques using machine learning.

The KDD 99 dataset [3] is the pioneer for machine learning based IDS. KDD 99 dataset was harvested from data gathering during the 1998 DARPA Intrusion Detection Evaluation Program, where a LAN was set up in an effort to simulate an actual military LAN, collecting TCPdump data over a duration of several weeks, with multiple attack data interspersed within normal connection data. The training data consists of five million connections records, and two weeks of testing data yielded around two million records. The training data contains 22 different attacks out of the 39 present in the test data. The known attack types are those present in the training dataset while the novel attacks are the additional attacks in the test datasets not available in the training dataset. The attacks types are grouped into four categories [3] :

- DOS (Denial of Service): Under this attack, the attacker prevents user from using resources by pre-occupying resources so that the service provider can no longer handle new user requests.
- Probing: Under the probing attack, an attacker gathers information to bypass existing security measures by port scanning.
- U2R (User to Root): Under this attack, attackers attempt to gain unauthorized access to local super user (root) privileges.
- R2L (Remote to Local): This attack means unauthorized access from a remote machine outside of the system to access a valid user account.

Despite of its potential, KDD 99 dataset is considered as having several drawbacks such as duplicate evidences and

inherent packet handling problem of TCPdump [3]. To bypass these limitations and create a dataset for better evaluation of machine learning based IDS techniques, an improved dataset known as NSL-KDD was created by removing all duplicate records. The inherent drawbacks in the KDD 99 dataset has affected detection accuracies of many IDSs. NSL-KDD contains essential records of the complete KDD 99 dataset. There is a collection of downloadable files at the convenience of the researchers. Three main refinements done on KDD 99 dataset were [3]:

- All of the redundancy issues were taken care of in order to enable the classifiers to provide un-biased classification results.
- A comparable number of train and test records were provided to conduct rational experiments
- The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD 99 dataset [3].

After their introduction, KDD 99 and NSL-KDD attracted many research efforts and numerous machine learning architectures were developed for these datasets. The analysis of these two datasets revealed that they are highly imbalanced [3], i.e., U2R attack class type has only a few samples whereas the normal type connection class has more than 50% presence. As a result, classifiers face difficulty to detect the minor classes. We did a literature survey on these two datasets and found that the testing accuracy of NSL-KDD considering all the features stuck at around 85% considering all attack types. So we were looking for the answer to the following question, **Is there any way to increase the test accuracy of NSL-KDD especially for the minor classes?**

Traditional imbalanced learning strategies include oversampling minority classes or undersampling majority classes [5]. Recently, authors in [6] introduced a novel method called 'few-shot learning', where they used a generic dataset to learn a feature representation using deep CNN structures. They learned new feature representations then went through a SVM classifier or 1-NN classifier for few-shot learning. The SVM or 1-NN classifier only needs few samples to predict the class label for new observations. They found that, since the feature extractor is optimized for the generic dataset but not the few-shot samples, this model can perform well on the generic dataset but not on the minor classes.

In this paper, we combine the traditional imbalance learning techniques: oversampling and undersampling [5] with the few-shot deep learning strategy [6] for few-shot intrusion detection. We extracted outputs from different layers of a deep CNN classifier and designed a SVM classifier and a 1-NN classifier for few-shot intrusion detection. We also considered oversampling and undersampling methods to make datasets balanced to compare their performances. We conducted a literature survey on these two datasets and to the best of our knowledge that extracting outputs from different layers of deep CNN classifier for few-shot learning is not yet implemented for both the KDD and NSL-KDD datasets.

Our contribution in this paper can be summarized as follows,

- We developed a few-shot deep learning method for intrusion detection, where the average class-wise performance was improved where data is highly imbalanced.
- We effectively incorporated the traditional imbalanced learning techniques, oversampling and undersampling, into few-shot deep learning for intrusion detection and achieved state-of-the-art performances on the two benchmark datasets.

The rest of the paper are organized as follows. Section II provides a brief literature review. We discussed classifiers used in this research in Section III. Pre-processing methods and numerical evaluation are presented in Section IV and Section V respectively. Finally, Section VI concludes this paper.

II. Related Works

Intrusion detection using KDD 99 and NSL-KDD is well studied in literature. Researchers proposed single, hybrid and ensemble classifiers to increase test accuracy. Niyaz in [7] proposed a deep learning approach (self taught learning TSL) with two stage classification. The approach consists of learning a good feature representation from unlabeled data and apply to labeled data for classification. The authors used sparse auto-encoder for unsupervised feature learning and soft-max regression for classification. Using NSL-KDD Dataset they obtained 79.1% test accuracy considering 5 classes.

Tang et al. [8] proposed a deep neural network model with 3 hidden layers for NSL-KDD. Using only 6 basic features , their model resulted into 75.75 % test accuracy. In [9] Zhang and Zulkermine proposed a random forest based anomaly detection model. Their hybrid framework combined misuse and anomaly detection. They evaluated it on KDD 99 data set. They converted the attacks into 2-class and got a detection rate of 94.7%. Naive Bayes classifiers were also used for intrusion detection problem in [10] and provided competitive results. The experiments were performed on KDD 99 data set and focused on 4 classes and 2 classes of attacks. The authors implemented a decision tree classifier along with the Naive Bayes. For 4 classes case, the model provided 91.28% and 91.47% by decision trees and Naive Bayes, respectively. For the 2-classes case, the decision trees yielded a classification accuracy of 93.02% while the Naive Bayes yielded a 91.45% accuracy.

Unsupervised learning techniques were also proposed for this problem. In [11], authors implemented a k-mean clustering for the NSL-KDD data set. In the paper, the author tried to classify data into the major attacks categories (4 clusters). the clustering algorithm provided a good distribution of data and showed that it is very useful for unlabeled data.

A number of papers applied feature selection methods to reduce the complexity of the data. Panda et al. [12] proposed a hybrid intelligent approach combining principal component analysis (PCA) and random forest among other techniques. In the paper, NSL-KDD data set was used with 2 classes. The system gave a high detection rate (almost 100%) and

TABLE I
CNN ARCHITECTURE FOR FEATURE EXTRACTION

Layer Index	Layer	Output Shape	Padding & Stride
1	Conv2D (64 filters, size: 1x3)	1, 38, 64	0,1
2	Conv2D (64 filters, size: 1x3)	1, 36, 64	0,1
3	Maxpooling2D	1, 18, 64	
4	Conv2D (128 filters, size: 1x3)	1, 16, 128	0,1
5	Conv2D (128 filters, size: 1x3)	1, 16, 128	1,1
6	Conv2D (128 filters, size: 1x3)	1, 16, 128	1,1
7	Maxpooling2D	1, 8, 128	
8	Conv2D (256 filters, size: 1x3)	1, 8, 256	1,1
9	Conv2D (256 filters, size: 1x3)	1, 8, 256	1,1
10	Conv2D (256 filters, size: 1x3)	1, 8, 256	1,1
11	Maxpooling2D	1, 4, 256	
12	Flatten	1024	
13	Fully Connected	100	
14	Dropout	100	
15	Fully Connected	20	
16	Softmax	5	

small false positive rates. Bouzida et al. [13] also used PCA for feature reduction and applied to KDD 99 data set. Then, the authors evaluated the proposed approach by using nearest neighbor (NN) algorithm and decision trees with/without PCA. The approach provided high accuracies for each attack class.

Mukkamala in [14] used SVM and Neural Networks (NN) (Multi-layer, feed forward network with 4 & 3 layers) for the intrusion detection problem. The models were evaluated on KDD 99 data set. Both SVM and NN delivered high accuracies (almost 100%) considering 2 classes. However, the evaluation showed in significant difference in training times between NN and SVM. Wun-Hwa Chen also used SVM and NN for the problem [15] and their approaches were evaluated on KDD 99 data sets considering 2 classes and both models provided very high accuracy rates (very close to 100%).

Authors in [17], proposed an optimizer to overcome the weakness of gradient based optimization used in deep learning algorithms. Their proposed optimizer controls both short-term knowledge within a task and long-term knowledge common among all the tasks. Experiments showed that their method performs better than natural baselines and is competitive to the state-of-the art in metric learning for few-shot learning. In [16], authors presented prototypical networks for few-shot learning algorithm. They trained these networks to specifically perform well in the few-shot setting by using episodic training. Apart from this, they demonstrated how to generalize prototypical networks to the zero-shot setting, and achieved state-of-the-art results on the CUB-200 dataset

In essence, a high variety of models were proposed for these two datasets. Nevertheless, the accuracy of NSL-KDD never reached beyond a certain point while considering all 41 features and 5 classes. This motivated us to look for another process other than traditional classifiers.

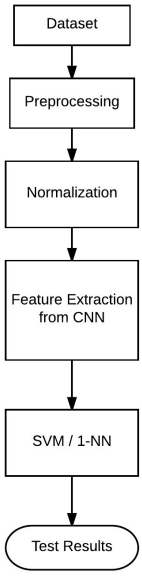


Fig. 1. Flowchart of our considered method.

III. CLASSIFIER INTRODUCTION

A. Support Vector Machine

SVM [18] is a supervised discriminative classifier which is known for detecting class labels by creating a separating hyperplane with large margins. In other words, given labeled training data, the algorithm provides an optimal hyperplane that maximizes the margins for all categories considered. SVM training algorithm builds a model that separates two categories with the largest margin that can generalize to unseen data. For multiple class problem, it usually utilize one-vs-all strategy to establish a multi-class classifier.

B. K Nearest Neighbor

k-NN [19] is a non-parametric learning algorithm, i.e., it does not consider any assumptions on data distributions. For a testing example, it searches k nearest neighbors of the testing example in the training dataset and assigns the majority of the labels from the nearest neighbors as the class label for the testing example. We utilized 1-NN classifier in this paper and it only needs a few training examples to predict class label for a new coming data point.

C. Convolutional Neural Network

CNN [20] is comprised of one or more convolutional layers (often with a maxpooling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. CNN achieved state-of-the-art performances in image classification and speech recognition fields in recent years. We utilized the CNN as a general feature extractor in this paper.

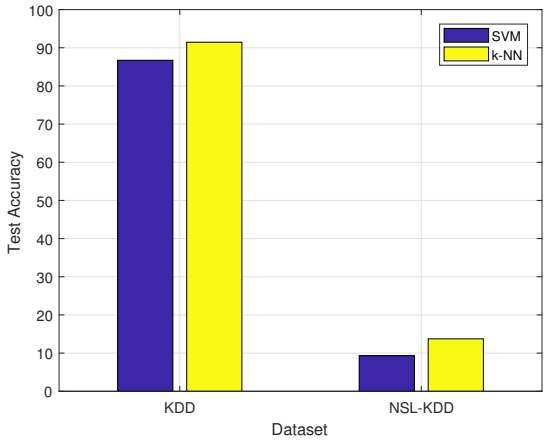


Fig. 2. Test Accuracy performance of undersampled datasets

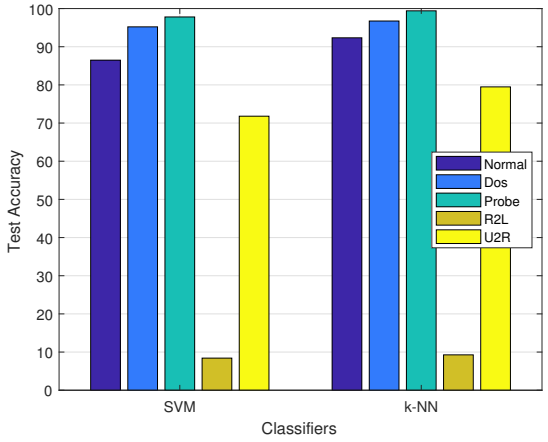


Fig. 3. Classwise Accuracy performance of undersampled KDD 99

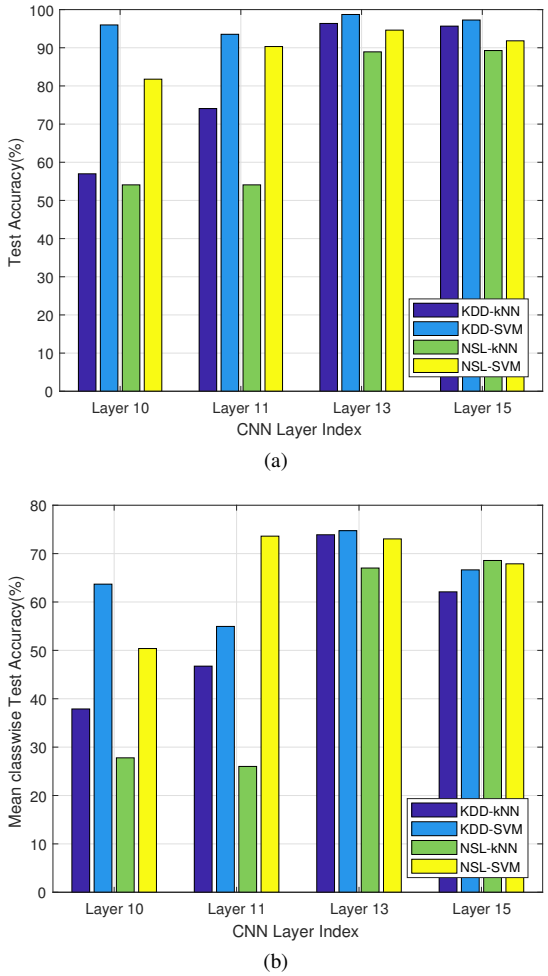


Fig. 4. (a) Test Accuracy performance & (b) Mean Class-wise test performance of features from different layers for original KDD and NSL-KDD datasets.

IV. Methods

A. Pre-processing

Neural network based classification uses only numerical values for training and testing. Hence a pre-processing stage is needed to convert the non-numerical values to numerical values. Two main tasks in our pre-processing are:

- Converting the non-numerical features in the dataset to numerical values. The features 2, 3 and 4 namely the protocol type, service and flag were non-numerical. These features in the train and test data set were converted to numerical types by assigning specific values to each variable (e.g. TCP = 1, UDP = 2 and ICMP = 3).
- Convert the attack types at the end of the dataset into its numeric categories. Category 1 is assigned to normal data, and 2, 3, 4 and 5 are assigned to DoS, Probe, R2L and U2R attack types, respectively

B. Normalization

Since the features of both KDD 99 and NSL-KDD datasets have either discrete or continuous values, the ranges of the values were different and this made them incomparable. In this study, the features were normalized by subtracting mean from each feature and dividing by its standard deviation. After that, we normalized the test features using the mean and standard deviation of each feature from train datasets.

C. CNN Architecture for Feature Extraction

We trained a CNN architecture to extract features for both datasets. Pre-processed data were fed through the input layer. We used various number of filters such as 64, 128 and 256 with filter size as 1x3. After the convolution layers, there was a fully connected dense neural network with 3 hidden layers with 100 , 20 and 5 hidden units respectively. We trained the model using train data and test data separately and extracted outputs from intermediate CNN layers to create new representations with different number of features. We considered mainly four layers in this study. The highest layer was a fully connected layer with 20 outputs, i.e., output from this layer has 20 features. We also considered a fully connected layer with 100 outputs, a maxpooling layer with 4x256 outputs and the last CNN layer which had 8x256 outputs. We also tried to extract features from lower level CNN layers but the testing accuracy was around 40% to 43% for both of the datasets and

TABLE II
ORIGINAL DATASET ACCURACIES

Dataset	Classifier	Test Accuracy
KDD 99	SVM	95.27%
	k-NN	96.19%
NSL-KDD	SVM	77.68%
	k-NN	80.74%

TABLE III
ORIGINAL DATASET CLASS-WISE MEAN ACCURACIES

Dataset	Classifier	Mean of Class-wise Test Accuracy
KDD 99	SVM	65.833%
	k-NN	64.048%
NSL-KDD	SVM	56.609%
	k-NN	53.84%

hence omitted for comparison. A brief overview of the CNN architecture is shown in Table I.

After getting the intermediate features we used them as input to SVM and k-NN. We considered 1 Neighbor for k-NN classifier. Fig. 1 shows our considered methodology and work-flow. As performance metric, we considered mean classwise accuracy along with test accuracy. In other words, we first calculated the accuracy for each class and then considered the mean of test accuracies of all classes as performance metric.

V. NUMERICAL RESULTS

A. Experiment Setup

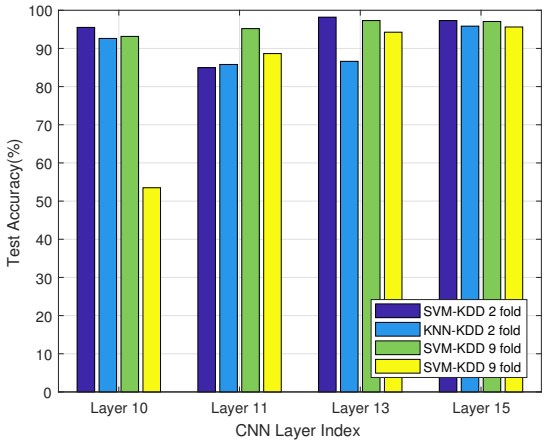
For model development and evaluation we have considered Intel core i-7 7700 3.60 Ghz CPU with 32 GB RAM work-station. We have implemented SVM using Liblinear [21] in MATLAB. For implementing k-NN we used the MATLAB built in function. The CNN was implemented using the Python keras package [22]. We have considered the following KDD and NSL-KDD datasets for this research:

- KDD Train : 10% KDDtrain
- KDD Test: Corrected_labels
- NSL-KDD Train : KDDTrain+
- NSL-KDD Test: KDDTest+

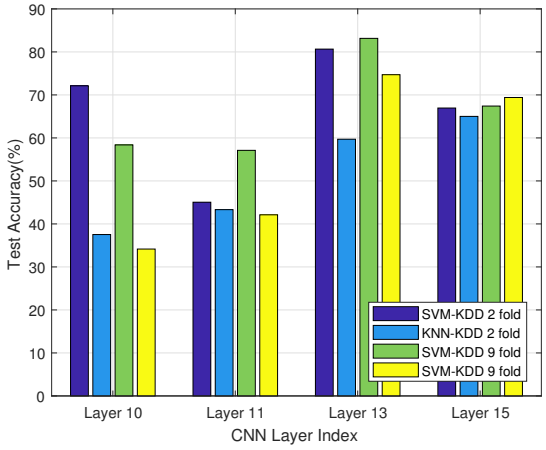
The 10 % KDDtrain dataset consisted of 494,021 records among which 97,277 (19.69%) were normal, 391,458 (79.24%) DOS, 4,107 (0.83%) Probe, 1,126 (0.23%) R2L and 52 (0.01%) U2R connections. In each connection, there are 41 attributes describing different features of the connection and a label assigned to each either as an attack type or as normal. The 41 attributes can be classified into four different categories as Basic, Content , Traffic and Host. Corrected_labels has 311027 records. NSL KDDTrain+ and KDDTest+ has 125973 and 22544 records respectively.

B. Initial Evaluation

We tried to measure the performance of the original datasets first. Table II shows the test accuracies for different classifiers for both datasets. As expected, the test accuracy for KDD 99 is much higher (about 96%) than the NSL-KDD for both of the classifiers due to its redundant records. The test accuracies for NSL-KDD is also close to those of previous literatures. Table III summarizes the class wise performance of the classifiers for both of the datasets. It turns out that KDD 99 dataset outperforms NSL-KDD in terms of detecting each class individually by scoring 65.83% and 64.05% for 1-NN and SVM respectively. For NSL-KDD, the classifiers were not able to detect the minor classes properly which resulted into class-wise performance degradation (53.84% and 56.609% respectively for 1-NN and SVM).



(a)

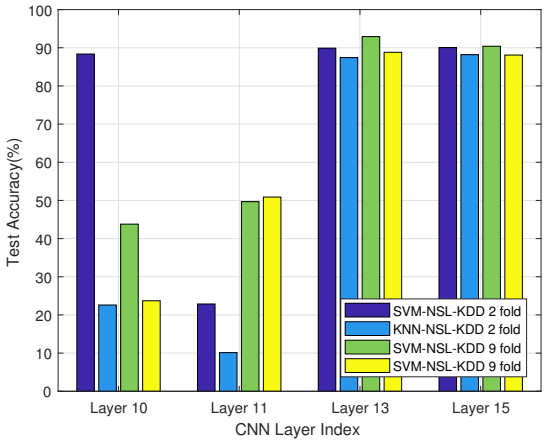


(b)

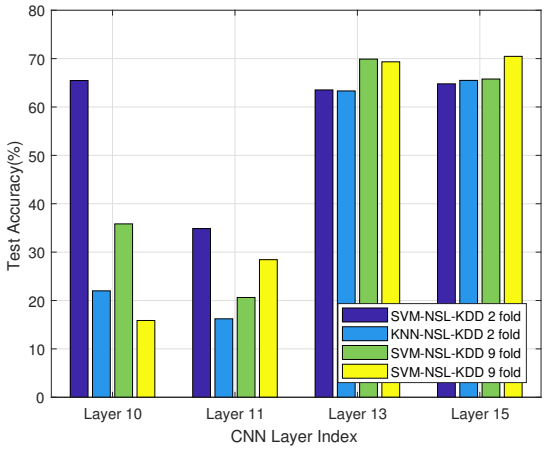
Fig. 5. (a) Test Accuracy performance & (b) Mean Class-wise accuracy performance of features from different layers for 2 & 9 fold oversampled KDD dataset.

C. Performances from Oversampling and Undersampling

As mentioned before, the U2R attack class has only .05% evidences. So deep learning methods are not able to classify this class correctly as they are biased towards more frequent classes. So we tried to undersample the other four classes randomly at the same evidence of U2R i.e., we tried to create new training datasets which is balanced (each class has 20% presence). Then we trained 2 classifiers (1-NN & SVM) using these datasets and compared the test performances using the original test datasets. The test accuracies of undersampled KDD 99 for the two classifiers were comparable to each other



(a)



(b)

Fig. 6. (a) Test Accuracy performance & (b) Mean Class-wise accuracy performance of features from different layers for 2 & 9 fold oversampled NSL-KDD dataset.

as depicted in Fig. 2. To mitigate the effect of randomness, we conducted the experiment 10 times and calculated the mean results. The accuracies for SVM and 1-NN were 91.66% and 87.3% respectively for KDD 99 dataset. But this undersampling method performed poorly on NSL-KDD dataset. The test accuracy reached merely highest only 13%. We were also curious about taking a look on class-wise performance of undersampled KDD 99. From Fig. 3, we can see the class-wise test performance of the classifiers. In this case, mean of the class-wise test accuracies were better than original datasets, where SVM and 1-NN scored 71.94% and 75.44 % respectively.

D. Performance of few-shot Deep Learning

We trained a CNN model using KDD 99 and NSL-KDD datasets and created 4 new datasets by extracting outputs from 4 different layers as mentioned in the previous section. The results are shown in Fig. 4. From Fig. 4(a), we can see that, the SVM test accuracies of KDD 99 for Layer 15 and Layer 13 are 97.26% and 98.71%, respectively, which are higher than original SVM considering 41 features as shown in Table II. The 1-NN accuracies are slightly better than the results as shown in Table II. For NSL-KDD the accuracies reached close to 90% for both of the classifiers. For instance, Layer 15 provides 91.82% for SVM and 89.27% for 1-NN. Features extracted from Layer 13 provided 94.62% for SVM classifier and 88.93% for 1-NN. The accuracies for NSL-KDD drop as we move to lower layers. We can also observe a similar pattern for mean class-wise test performances of different CNN layers in Fig. 4(b). The more we go down, the less we get mean test accuracy of all the classes. The best performance is provided by layer 13 where the class-wise test accuracies of all classifiers and datasets were above 70%. The SVM classifier on KDD 99 dataset provided better results than other classifiers.

E. Effect of Sampling on Low shot Deep Learning

To increase the class-wise performance of original datasets, we created 2 fold and 9 fold duplicate samples of U2R class and studied the performances for both datasets. The results for 2 & 9 fold duplicate oversampling of U2R class on KDD dataset is depicted in Fig. 5. From Fig. 5(a) we can see that the testing performances of 2 fold oversampled KDD 99 dataset for SVM classifier were 97.29%, 98.19%, 84.96% and 95.51% respectively for Layers 15,13,11 and 10. The 1-NN classifier performance on same oversampled dataset were 95.84%, 86.62%, 85.8% and 92.62%, respectively. In case of, 9 fold oversampling of class U2R, KDD provided 97.06 %, 97.3%, 95.19% and 93.152% testing accuracies for SVM classifier. On the other hand, 1-NN scored 95.62%, 94.25%, 88.65% and 53.5% on the same oversampled KDD dataset. We then studied mean class-wise accuracies for 2 fold and 9 fold duplicate evidences of U2R class on KDD which is depicted in Fig. 5(b). The best results are provided by the features from Layer 13. Overall, the mean class-wise performance is better than the original dataset. We also observed that, the performance of 9 fold oversampled outperformed its 2 fold counterpart for both Layer 13 and 15. SVM classifier scores 83.152% mean class-wise accuracy on features extracted from Layer 13 for 9 fold U2R oversampled KDD.

The test performances for 2 & 9 fold duplicate evidences of U2R class on NSL-KDD dataset is depicted in Fig. 6. The testing accuracies of 2 fold oversampled NSL-KDD dataset for SVM classifier were 90.043%, 89.9%, 22.87% and 88.36% respectively for layers 15,13,11 and 10 as shown in Fig. 6(a). The test accuracies of 1-NN classifier performance on same oversampled dataset were 88.19%, 87.42%, 10.13% and 22.60%, respectively. The performance of 9 fold oversampled dataset is slightly better than 2 fold oversampled NSL-KDD. In case of, 9 fold oversampling of class U2R, NSL-KDD provided 90.4%, 92.92%, 49.67%, 43.79% testing accuracies for SVM classifier. At the same time, 1-NN scored 88.09%, 88.82%, 50.88% and 23.71% accuracies on the same oversampled dataset for features from layer 15 to 10.

The mean classification accuracies of NSL-KDD are shown in Fig. 6(b). We found a decreasing pattern for mean class-wise test accuracy as we move downwards to CNN architecture.

TABLE IV
TEST ACCURACY COMPARISON TO LITERATURE

Algorithms	Test Accuracy
Niyaz ei al. [7]	79.10%
Mahbod et al. [3]	82.02%
Tang et al. [8]	75.75%
Our work	**94.62**%

The highest mean class-wise accuracy (70.46%) was provided by the SVM classifier for 9 fold oversampled U2R attack class. As we move down towards the architecture, the mean class-wise accuracy decreases. In essence, we observed an increasing trend in mean class-wise performance but a slight degradation in test accuracy performance for both of the oversampled datasets. This is due to the fact that the classifiers are detecting the minor class more accurately at the expense of compromising test accuracy performances of majority classes.

F. Comparison to Literature

For ease of comparison with previous literatures considering 4 attack types, we also provide Table IV which shows that our method outperforms other results in terms of test accuracy. SVM classifier on features extracted from layer 13 provided the best result on NSL-KDD dataset. Our methods worked well for both of the datasets in terms of overall test accuracy and class-wise test accuracy. The Layer 13, which is the first fully connected layer with 100 hidden units, provided the best results. Among the two classifiers, SVM outperformed 1-NN in almost all the experiments.

VI. CONCLUSION

In this research, we implemented a few-shot deep learning method for intrusion detection. Among different attack types, some rare attack types make machine learning based detection systems difficult to identify those minority attack types. Inspired by the few-shot image recognition work in [6], we trained a deep CNN structure and used it as a general feature extractor for feature extraction. We then trained a SVM or an 1-NN classifier for intrusion detection on the new feature representations. In addition, we incorporated a traditional imbalance learning technique that oversampled minority classes before training. Our method obtained state-of-the-art performances on the KDD and NSL-KDD datasets achieving over 94% accuracies for both datasets. We also able to achieve better classwise accuracy using traditional imbalance learning techniques. The proposed method is a good candidate for imbalance learning and intrusion detection. In future, we plan to use our method on various imbalanced datasets to enhance the minority class detection rate.

REFERENCES

[1] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sept 2016, pp. 1–8.

[2] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, "Selecting features for intrusion detection: A feature relevance analysis on kdd 99 intrusion detection datasets," in *Proceedings of the third annual conference on privacy, security and trust*, 2005.

[3] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, July 2009, pp. 1–6.

[4] J. P. T. Srilatha Chebrolu, Ajith Abraham, "Feature deduction and ensemble design of intrusion detection systems, computers security, volume 24, issue 4, june 2005, pages 295-307, issn 0167-4048."

[5] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, Sept 2009.

[6] B. Hariharan and R. Girshick, "Low-shot visual recognition by shrinking and hallucinating features," *arXiv preprint arXiv:1606.02819*, 2016.

[7] Q. Niyaz, W. Sun, A. Y. Javaid, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS), BICT-15*, vol. 15, 2015, pp. 21–26.

[8] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, Oct 2016, pp. 258–263.

[9] J. Zhang and M. Zulkernine, "A hybrid network intrusion detection technique using random forests," in *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*. IEEE, 2006, pp. 8–pp.

[10] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive bayes vs decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM symposium on Applied computing*. ACM, 2004, pp. 420–424.

[11] V. Kumar, H. Chauhan, and D. Panwar, "K-means clustering approach to analyze nsl-kdd intrusion detection dataset," *International Journal of Soft Computing and Engineering (IJSCE)*, 2013.

[12] M. Panda, A. Abraham, and M. R. Patra, "A hybrid intelligent approach for network intrusion detection," *Procedia Engineering*, vol. 30, pp. 1–9, 2012.

[13] Y. Bouzida, F. Cuppens, N. Cuppens-Boulahia, and S. Gombault, "Efficient intrusion detection using principal component analysis," in *3éme Conférence sur la Sécurité et Architectures Réseaux (SAR), La Londe, France*, 2004, pp. 381–395.

[14] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, vol. 2. IEEE, 2002, pp. 1702–1707.

[15] W.-H. Chen, S.-H. Hsu, and H.-P. Shen, "Application of svm and ann for intrusion detection," *Computers & Operations Research*, vol. 32, no. 10, pp. 2617–2634, 2005.

[16] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," *CoRR*, vol. abs/1703.05175, 2017. [Online]. Available: http://arxiv.org/abs/1703.05175

[17] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[18] D. S. Kim, H.-N. Nguyen, and J. S. Park, "Genetic algorithm to improve svm based network intrusion detection system," in *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, vol. 2, March 2005, pp. 155–158 vol.2.

[19] H.-V. Nguyen and Y. Choi, "Proactive detection of ddos attacks utilizing k-nn classifier in an anti-ddos framework," *International Journal of Electrical, Computer, and Systems Engineering*, vol. 4, no. 4, pp. 247–252, 2010.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[21] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442794

[22] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.