Deep Learning for Network Intrusion Detection

Zhikai Chen, Xinyu Cai, Zhipeng Huang
School of Information Science and Technology
University of Science and Technology of China
{czk654, cxyvlf, hzp1104}@mail.ustc.edu.cn

***Abstract*—Network Intrusion Detection System (NIDS) monitors the running state of a network to detect attacking attempts, behaviors and results, for the sake of protecting the network system's confidentiality, integrity and availability.**

In this paper, we focus on intrusion detection system in the network security instead of computer (host) security and summarize the recent deep learning methods in particular for characteristics of internet environment, then propose our assumed approaches and rising prospects for further research in Deep Learning for Cyber-security.

I. Introduction

Cyber-security is the practice of protecting systems, networks, and programs from digital attacks. These cyber-attacks are usually aimed at accessing, changing, or destroying sensitive information; extorting money from users; or interrupting normal business processes. Implementing effective cyber-security measures is particularly challenging today because there are more devices than people, and attackers are becoming more innovative. Intrusion detection system (IDS) plays a necessary role in network security in order to discover, determine, and identify unauthorized usage, duplication, alteration, and destruction of information systems[1]. By collecting key information from a computer network and analyzing it, it is possible to discover whether there are violations of security policies and signs of attacks in the networks.

There are three main types of cyber analysis in support of NIDS: misuse-based, anomaly-based and hybrid. Misused-based methods aim at collecting the illegal operations in networks and building a database for attacks, then it is able to match the characteristics of known attacks automatically for behaviors to detect intrusion among them. However, they are not pretty appropriate especially for some complex environments of networks nowadays partly because the frequent manual updates of database and the lack of intrusion connections, which limits the capability of these methods to capture their features. Anomaly detection techniques such as Isolation Forest can handle the imbalanced data, which uses plenty of normal network activities to model the profiles of normal network. With their ability to detect novel attacks and existed datasets, they become popular with learning-based methods recently since detection can be regarded as a classification problem for either raw traffic data, packets or their features.

From the perspective of artificial intelligence (AI) development, traditional port-based and deep packets inspection (DPI)-based methods are rule-based approaches, which perform classification by matching predefined hard-coded rules. Statistical-based and behavioral-based methods are classic machine learning (ML) approaches, which classify traffic by extracting patterns from empirical data using a set of selective features.

A simple definition states that Deep Learning (DL) is a set of machine learning algorithms that attempt to learn in multiple levels, corresponding to different levels of abstraction. The levels correspond to distinct levels of concepts, where higher-level concepts are defined from and helped lower-level ones. Feature extraction is performed by the first few layers of the deep network. There are unsupervised, supervised, and hybrid DL architectures. Because shallow neural networks have only one hidden layer, they lack the ability to perform advanced and unable to learn high-level concepts as deep neural networks. This also holds true for other machine learning algorithms, as well. Deep learning algorithms well capture the intrusion features and packets data distribution and it becomes possible to update automatically the model's parameters according to the situation in complex networks environment while there are unknown data packets. Further more, these methods usually need no hand-designed features but directly took raw traffic as input data of classifier which is representation learning different from rule-based or classic ML approaches as shown in Fig. 1 As the development of effective DL methods and collection of various data set, it has achieved state-of-the-art performance in different experiments and derived a series of network intrusion detection system.

This work can be generally divided into three parts. (i) We introduce the datasets generally used in deep learning methods for intrusion detection and the metrics to evaluate the discriminate model. (ii) We illustrate the thesis and significant components of recent influential works in this field, including deep belief networks (DBN), autoencoder or variant autoencoder (VAE), convolution neural networks (CNN), recurrent neural networks (RNN) or long-short term memory networks (LSTM) and generative adversarial networks (GAN) utilized in intrusion detection for different application. (iii) We assume some methods in two different perspectives for future improvement, including a means to improve the present limitation such as high false detection rate of anomaly detection and imbalanced data for some attacking classes and an application of a potential and feasible framework for intrusion detection in a series of

data packets based on long-term recurrent convolutional networks with images analysis.

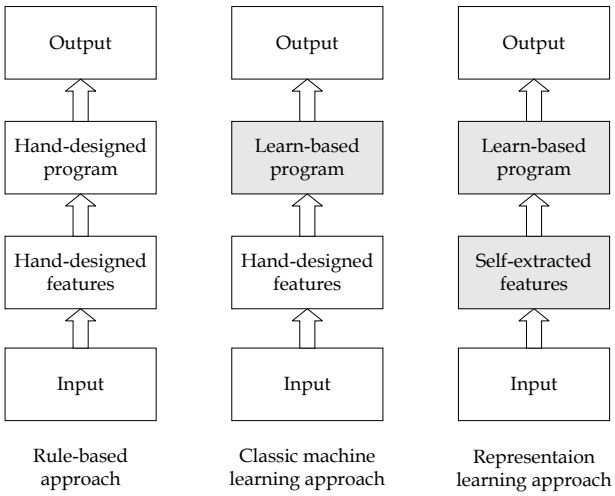

Fig. 1. Work flows of traffic classification

II. Evaluation

A. Metrics

This section describes several classification metrics used by traditional valuation. For a model performing a binary classification task, there are a number of different metrics. Accuracy, precision, recall, false positive rate, F1 Score, and area under curve (AUC) are included and many of the metrics have more than one name. All of these evaluation metrics are derived from the four values found in the confusion matrix (Table I), which is based on the calculated predicted class versus the actual class (ground truth).

TABLE I
Confusion matrix

Actual \ Predicted	Malicious	Benign
Malicious	True Positive (TP)	False Negative (FN)
Benign	False Positive (FP)	True Negative (TN)

Accuracy (acc) or Proportion Correct: the ratio of correctly classified examples to all items.The usefulness of accuracy is lower when the classes are unbalanced (i.e., there are a significantly larger number of examples from one class than from another). However, it does provide useful insight when the classes are balanced.

$$acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Positive Predictive Value (PPV) or Precision (p): The ratio of items correctly classified as class X total items that were classified as class X

$$p = \frac{TP}{TP + FP} \tag{2}$$

Sensitivity or True Positive Rate (TPR) or Probability of Detection(PD)or Recall (r): The ratio of items correctly classified as X to all items that were actually class X.

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

Negative Predictive Value (NPV): The ratio of items correctly classified as not X to all items classified as not X.

$$NPV = \frac{TN}{TN + FN} \tag{4}$$

Specificity or True Negative Rate (TNR): The ratio of items correctly classified as not X to all items that are not class X.

$$TNR = \frac{TN}{TN + FP} \tag{5}$$

False Alarm Rate (FAR) or False Positive Rate (FPR) or Fall-Out: The ratio of items incorrectly classified as class X to all the items that are not class X.

$$FPR = \frac{FP}{TN + FP} \tag{6}$$

F1 Score (F1): The F1 Score is the harmonic mean of the precision (p) and the true positive rate (r).

$$F_1 = \frac{2}{\frac{1}{r} + \frac{1}{p}} = 2\frac{p * r}{p + r} \tag{7}$$

This is a specific version of the F-βfunction, in which precision and true positive rate are given equal importance.

Area under the curve (AUC): The sum of the area under a receiver operating characteristic (ROC) curve, which is a plot of the false positive rate versus the true positive rate, created by varying the classification thresholds.

For multi-class problems, accuracy can be easily calculated; however, metrics such as precision, recall, FPR, F1 Score, and AUC cannot be calculated in a straightforward fashion (e.g., TP, TN do not exist for three-class problems). Precision, recall, etc. can be determined for a 3+ class problem by collapsing the problem into a two-class problem (i.e., all versus one), where the metrics are calculated for each class. Usually, the only accuracy is used for multiclass problems.

It is important to remember that because each of the papers described later uses a different dataset (or sometimes a different subset of a given dataset), it is not possible to compare the models developed based on the accuracy (or any other metrics) they obtained. Such a comparison would be valid only if the authors of both publications used exactly the same training dataset and the same testing dataset.

B. Related Datasets

An overview of some widely-used and significant datasets are shown in Fig. II. Then we give an introduction of them one by one as follows.

TABLE II
ATTACKS WITHIN THE NETWORK-BASED DATA SETS

Data Set	Attacks
CIDDS-001	DoS, port scans (ping-scan, SYN-Scan), SSHbrute force
CIDDS-002	port scans (ACK-Scan, FIN-Scan, ping-Scan,UDP-Scan, SYN-Scan)
CTU-13	botnets (Menti, Murlo, Neris, NSIS, Rbot, Sogou,Virut)
DARPA	DoS, remote-to-local, user-to-root, probing
KDD CUP 99	DoS, remote-to-local, user-to-root, probing
NSL-KDD	DoS, remote-to-local, user-to-root, probing

1) DARPA 1998/1999: The Defense Advanced Research Projects Agency (DARPA) 1998 [1] and DARPA 1999 data sets [2] are extensively used in experiments and frequently cited in publications. The DARPA 1998 set was created by the Cyber Systems and Technology Group of the Massachusetts Institute of Technology Lincoln Laboratory (MIT/LL). DARPA 1999 data set had substantially more attack types than the DARPA 1998 data set. In both collections, the data sets were processed and curated to be used in the experiments. The TCP dumps and logs were combined into one stream with many columns. The data in DARPA dataset are tcp dump. Training data and test data are recorded for 5 million connections and 2 million connections. The dataset has 41 features such as protocol type, flag, duration, service etc and their descriptions. And 22 attacks such as DoS, Probe, U2R, R2L etc. are recorded in the dataset. DoS means denial-of-service. Probe is a surveillance attack. U2R is an attack which tries to unauthorized access to superuser. R2L is an unauthorized remote access attack.

2) KDD 1999: The Knowledge Discovery and Dissemination (KDD) 1999 dataset analyzed by [3] is one of the most widely used datasets for intrusion detection with about 4 million records of normal and attack traffic. There are a huge number of redundant records (78% in the training data and 75% in test data) causing bias. In addition, in the classification experiments the group conducted, they pointed out that by randomly selecting subsets of the training and testing data, often very high, unrealistic accuracies can be achieved. Therefore, Tavallaee et al. proposed a new data set, NSL-KDD, in order to overcome this shortcomings.

3) CIDDS 001/002: CIDDS (Coburg Intrusion Detection Data Sets) is a concept to create evaluation data sets for anomaly-based network intrusion detection systems. The CIDDS-001 data set was captured within an emulated small business environment in 2017, contains four weeks of unidirectional flow-based network traffic, and comes along with a detailed technical report with additional information. As a special feature, the data set encompasses an external server which was attacked in the internet.

CIDDS-002 is a port scan data set which is created based on the scripts of CIDDS-001. The data set contains two weeks of unidirectional flow-based network traffic within an emulated small business environment. CIDDS-002 contains normal user behavior as well as a wide range of different port scan attacks. A technical report provides additional meta information about the data set where external IP addresses are anonymized.

4) CTU-13: CTU-13 is the most commonly used dataset which contains raw packet data. The benefit of raw pcap files is the opportunity for individuals to perform their own preprocessing, enabling a wider range of algorithms to be used. Additionally, the CTU-13 is not a simulated dataset. It contains 13 different scenarios with different numbers of computers and seven different botnet families. CTU-13 dataset and IXIA dataset compose USTC-TFC 2016 which contains 10 types of malicious data and 10 types of normal data. USTC-TFC 2016 is used to a CNN-based NIDS which will be illustrated in Section III-D.

III. Recent Works

Network intrusion detection systems are essential for ensuring the security of a network from various types of security breaches. There have been many approaches to intrusion detection using DL. We will explore them as follows.

A. Deep Belief Networks

Hinton et al. introduced Deep Belief Networks (DBNs) based on Restricted Boltzmann Machine (RBM) in [4]. An RBM consists of the visible units and hidden units. The structure is shown in Fig. 2. We use the vector v and h represent the visible units and the hidden unit state respectively. Then v_i denotes the state of the i visible unit, h_j denotes the state of the j hidden unit, w, a_i, b_j are the weights between them and their bias respectively. They meet the energy formula 8 as follows. With this structure, RBM can be trained in an unsupervised manner just with input for encoding or decoding. As a result, the weight updating rule as shown in formula 9 go through a bidirectional propagation, where $\langle v_i h_j \rangle^+$ indicates the average forward correlation and $\langle v_i h_j \rangle^-$ indicates the average reverse correlation. ϵ denotes the learning rate.

$$E(v,h) = -\sum_{i \in V} b_i v_i - \sum_{j \in H} a_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{8}$$

$$\begin{aligned} w_{ij}(t+1) &= w_{ij}(t) + \Delta w_{ij} \\ &= w_{ij}(t) + \varepsilon\left(\langle v_i h_j \rangle^+ - \langle v_i h_j \rangle^-\right) \end{aligned} \tag{9}$$

DBNs are a class of deep neural networks composed of multiple layers of RBMs mentioned above and followed with a back propagation layers (BP) which can be a classification

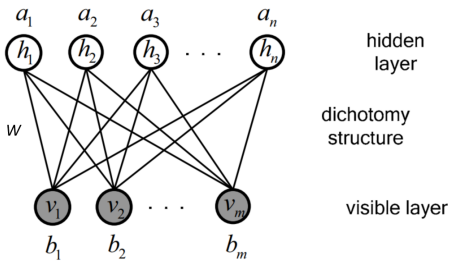

Fig. 2. typical Restricted Boltzmann Machine

layer. RBMs are trained in an unsupervised manner. Typically each RBM layer in DBN is trained in unsupervised manner one after another, and finally train BP layer using back propagation in supervised manner.

Many network gateways and routers devices, which could potentially host a NIDS, do not have enough memory or processing power to train and sometimes even execute such model. Additionally, used in various situations and to detect numerous abnormal data packets, the parameters in a pretrained NIDS often need to be fine-tuned with current real-time data after deployed to the network nodes.

A drawback of pure BP neural networks is that it needs a great amount of resources to train themselves, once begin training, the whole model is adjusted dramaticly.

Unlike traditional neural networks, DBNs' training process can be divided into two parts, pre-training and fine-tuning. Pre-training is processed in GPU and it generates an effective deep RBM which can refactor data and reduce data dimensions. After that, a two-level back-propagation network is added to it. The fine-tuning process on the network node only needs to adjust the last simple back-progradation network, which is a light, efficient and practical online manner and able to update the system from time to time.

Gao et al. used a DBN for intrusion detection in [5] with KDD 1999 dataset. The best performing algorithm was a DBN with four hidden layers (six layers total), beating an SVM and DBNs with fewer layers. The accuracy was 93.49%, with a TPR of 92.33%. Nguyen et al. [6] achieved similar accuracy using a similar architecture. Alrawashdeh and Purdy [7] performed a similar experiment with a four-hidden-layer DBN and achieved an accuracy of 97.9%. Alom et al. [8] built a similar model, but were able to achieve 97.5% accuracy,training on 4% of the data.

B. Deep Autoencoders

Autoencoders are a class of unsupervised neural networks composed of an encoder and a following decoder in which the network takes a vector as input and tries to match the output to that input vector. In another word, encoder is used to reconstruct the input to a hidden vector, and the decoder can recover the input vector from the hidden vector. Thus, the encoder can be used for dimension reduction by encoding or reconstructing the input, creating a higher

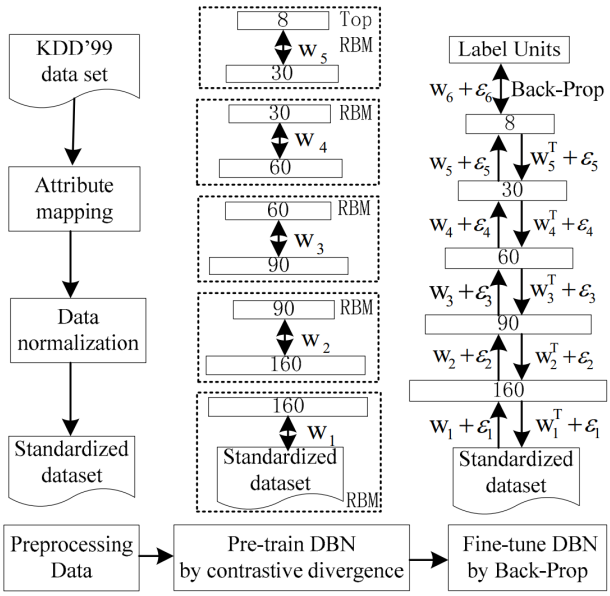

Fig. 3. Deep Belief Network

or lower dimensionality representation of data. Typically, autoencoders are applied to extract the features of raw data.

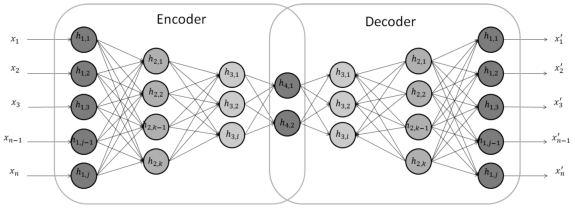

Fig. 4. Autoencoder

Dimensionality reduction to higher-level features using autoencoders or RBMs with unsupervised extreme machine learning obviously promote the training process of subsequent K-means clustering or classifying for normal/anomaly packets, which is able to fuse semantic information.

Li, Ma, and Jiao [9] first used an autoencoder to reduce the dimensionality of the data, followed by a DBN with RBM layers that achieved a TPR of 92.2% with a FPR of 1.58%. Yousefi-Azar el al. [10] used an autoencoder with four hidden layers, followed by a Gaussian naive Bayes classifier and achieved an accuracy of 83.34%. Alom and Taha [11] implemented both autoencoders and RBMs to perform dimensionality reduction on the KDD-1999 dataset, reducing it to nine features, and then performed K-means clustering on the data, achieving detection accuracies of 91.86% and 92.12% accuracy, respectively.

C. Variant Auto-Encoder

Variant Auto-Encoder (VAE) is a specific type of autoencoder but has different purpose. Autoencoder is usually

used to extract features by utilizing the encoder part. However, VAE is a generative model which is used to generate data. In VAE, the latent vectors that encoder produces are restricted to a certain distribution such as normal distribution, so that given a sample of normal distribution, the decoder can generate a data in real space. This structure is shown in Fig. 5.

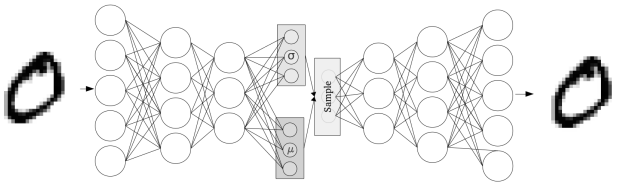

Fig. 5. Variant Auto-Encoder

As is shown in Fig. 5, the output of encoder is restricted to normal distribution (with mean μ and standard deviation σ) close to standard normal distribution by minimize formula 10. BP can not work because of the sampling operation between encoder and decoder, VAE solves this using reparameterization trick. Once the training finishes, the decoder can be used to generate real data with a sample from standard normal distribution as input.

$$KL\left(N\left(\mu,\sigma^2\right)\|N(0,1)\right)=\frac{1}{2}\left(\mu^2+\sigma^2-\log\sigma^2-1\right) \qquad (10)$$

As imbalanced dataset, in which anomaly data is far less than normal data, limits the capability of learning methods, synthetic minority reconstruction technique (SMRT) is proposed which uses VAE as a classifier with only the original class distribution. The role of the VAE is to generate synthetic observations of the attack class, which is the minority class, for the sake of handling imbalanced data.

Using the CIDDS-001, Abdulhammed et al. [12] trained a variational autoencoder, a specific type of autoencoder, as well as other machine learning algorithms, to perform intrusion detection, which achieves 97.59% accuracy, lower than some of the other methods, which were trained using class imbalance correction techniques. The best method was majority voting classifiers, which achieved 99.99% accuracy. Mirsky et al. [13] created an ensemble of autoencoders, ranging in number from 2 to 48, that were tasked with reconstructing subsets of features, determined using clustering with correlation as the distance metric. The root-mean-square errors (RMSE) of reconstruction from each autoencoder is then used to train a new autoencoder, in which the RMSE represents the anomaly score. Mirsky et al. found that their algorithm performed comparably to or better than algorithms such as isolation forests and Gaussian mixture models.

D. Convolutional Neural Networks

Convolutional neural networks (CNNs) play an important role in recent breakthroughs in computer vision. CNNs are neural networks meant to process input stored in arrays. An example input is an image, which is a 2-dimensional(2D) array of pixels. The architecture of a CNN consists of three distinct types of layers: convolution layers, pooling layers, and the classificaiton layers.

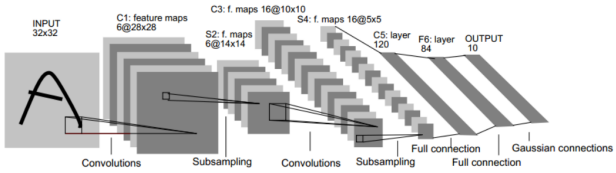

Fig. 6. Convolutional Neural Network

Wang et al. [14] built an intrusion detection algorithm using raw network traffic data from two existing datasets: the CTU-13 dataset and the IXIA dataset (that the authors called the USTC-TFC2016 dataset), which contained 10 types of normal data and 10 types of malicious data, and appeared to be relatively balanced between malicious and normal. A preprocessing step took the raw network traffic data and converted it into images, which were then fed into a CNN with a similar architecture to the well-established CNN LeNet-5 (LeCun et al. [15]). Because there was no engineering of the preprocessing stage that produced the images, this method handled the raw data directly. The classification was done in two different ways. The first method involved a 20-class classifier, and the goal was to identify which type of normal or malicious the traffic was. The second was a binary classifier which fed into one of two CNNs trained to identify the type of malicious traffic or binary traffic. The 20-class classifier achieved an accuracy of 99.17%. The binary classifier achieved 100% whereas the 10-class normal classifier achieved 99.4% and the 10-class malicious classifier achieved 98.52%.

E. Recurrent Neural Networks

Recurrent neural network (RNNs), extends the capabilities of a traditional neural network, which can only take fixed-length data inputs, to handle input sequences of variable lengths. The RNN can processes inputs one element at a time, using the output of the hidden units as additional input for the next element. Therefore, the RNNs can time series problems such as cyber traffic data.

Standard RNN cannot bridge more than 5–10 time steps. Error signals tend to either blow-up or vanish. Blown-up error signals lead straight to oscillating weights, whereas with a vanishing error, learning takes an unacceptable amount of time, or does not work at all. A detailed theoretical analysis of the problem with long-term dependencies is presented in [16]. The paper also briefly outlines several proposals on how to address this problem.

One solution that addresses the vanishing error problem is a gradient-based method called long short-term memory(LSTM) published by [17][18][19]. LSTM can learn how to bridge minimal time lags of more than 1,000

discrete time steps [17]. The solution uses constant error carousels(CECs), which enforce a constant error flow within special cells. Access to the cells is handled by multiplicative gate units, which learn when to grant access. In the absence of new inputs to the cell, we know that the CEC's backflow remains constant. However, as part of a neural network, the CEC is not only connected to itself, but also to other units in the neural network. We need to take these additional weighted inputs and outputs into account. Incoming connections to neuron j can have conflicting weight update signals, because the same weight is used for storing and ignoring inputs. For weighted output connections from neuron j, the same weights can be used to both retrieve j's contents and prevent j's output flow to other neuron s in the network. To address the problem of conflicting weight updates, LSTM extends the CEC with input and output gates connected to the network input layer and to other memory cells. This results in a more complex LSTM unit, called a memory cell; its standard architecture is shown in Fig 7.

Research results [20] show that the LSTM classifier provides superior performance in comparison to other tested strong static classifiers. The LSTM classifier shows its strength when training 'DoS' attacks and network probes. The target neuron representing DoS attacks even achieves close to perfect discrimination between attacks and other traffic. These traffic classes tend to generate a high volume of consecutive connection records. Here, LSTM can strongly benefit from the fact that it can look back in time and learn to correlate these connections. So LSTM is very suitable for classifying high-frequency attacks.

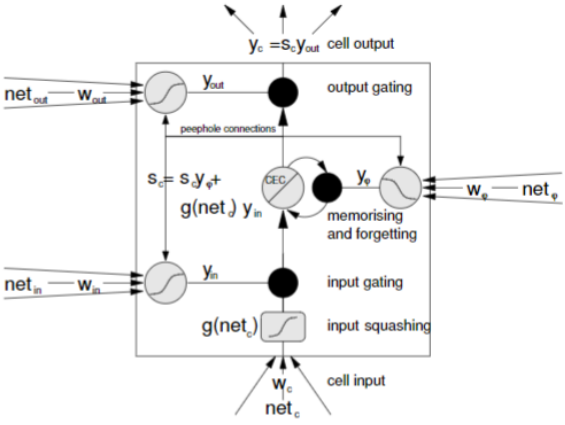

Fig. 7. standard LSTM memory cell

F. Generative Adversarial Networks

Generative adversarial networks (GANs), are a type of neural network architecture used in unsupervised machine learning, in which two neural networks compete against each other in a zero-sum game to outsmart each other. One network acts as a generator and another network acts as a discriminator. The generator takes in input data and generates output data with the same characteristics as real data. The discriminator takes in real data and data from the generator and tries to distinguish whether the input is real or fake. When training has finished, the generator is capable of generating new data that is not distinguishable from real data. The generator can learn the distribution of the real data.

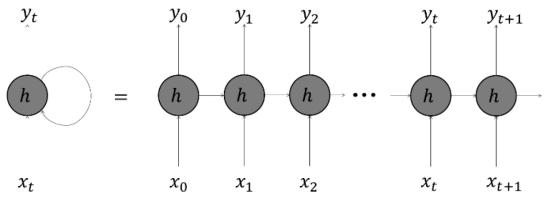

Fig. 8. Recurrent Neural Network

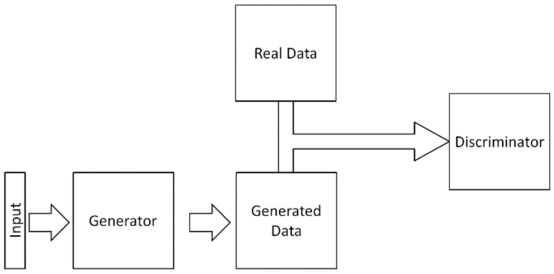

Fig. 9. Generative Adversarial Network

In most datasets, anomaly data is far less than normal data. Many methods were proposed to balance the dataset, such as the VAE-based method mentioned in Section III-C, but GAN-based methods shows a different approach. A GAN can be trained with normal data, so that it learns the distribution of normal data, and the discriminator can be used as a classifier to distinguish anomaly data from normal. AnoGAN[21] is proposed to better extract the feature of normal samples, by establishing a mapping between the real space and latent space. Furthermore, intermediate layer was introduced in discriminator to optimize the feature extraction. But this mapping is based on the back-propagation algorithm, thus when the dimension of data increases, this model will be timeconsuming and not suitable for timely intrusion detection. To ease the above challenge, a new model [22] is adopted, inspired by the structure of BiGAN[23], to remarkably reduce the time cost.

However, the following problems remains in the anomaly detection task. In the training data for cyber-intrusion detection, those discrete features are lethal to traditional GANs, during whose training process the loss criterion is cross-entropy – a measurement that needs the continuity of features. To overcome the above hurdles, Chen et al.[24] proposed a GAN-based model with refined loss function to obtain an outstanding performance on the imbalanced

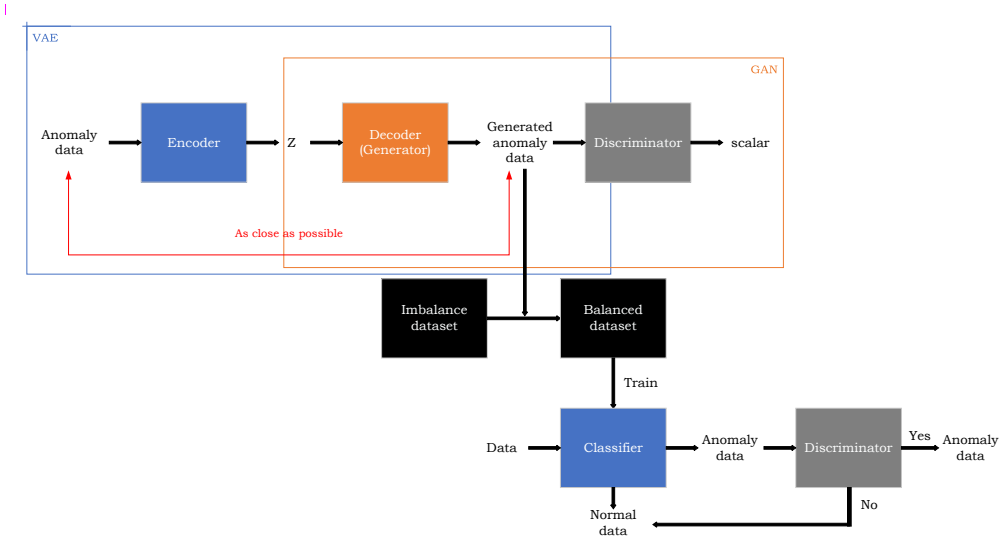

Fig. 10. VAE-GAN-based

dataset with discrete features. Furthermore, we used the multiple intermediate layers to extract the features in more complex environment.

IV. Prospects

A. LRCN

In host system, anomaly traffic can be a serious problem. Suppose a scenario that attackers mix some malicious data into a series of normal packets in a period of time when the terminal is receiving data. Then how to recognize an anomaly packet in a series of packets is really important for system protection. In order to combine temporal information or connections between each packet into the independent features/distribution of one packet, we assume one mechanism to achieve this point. Under this assumption, we can transform a series of data packets into a set of video frames by image generation from traffic data and regard them as a piece of action from outside networks on host.

Deep convolutional neural networks have achieved great success for visual recognition in still images. However, for action recognition in videos, the advantage over traditional methods is not so evident. We aims to design effective ConvNet architectures for action recognition in videos and learn these models given limited training samples by introducing temporal segmentation.

One succinct preprocessing measure is as follows: First trims or pads all data files to uniform length such as 784 bytes. Then the result can be converted to gray images for every byte range from 0x00 (0) to 0xff (255) representing a pixel. After that they can be analyzed in a visual way. The long-term recurrent convolutional networks (LRCN) framework for proposed by Donahue et al. [25] video-based action recognition is based on the idea of long-range temporal structure modeling as shown in Fig. 11 which combines a sparse temporal sampling strategy and frame series-level supervision to enable efficient and effective learning using the whole normal/intrusion action.

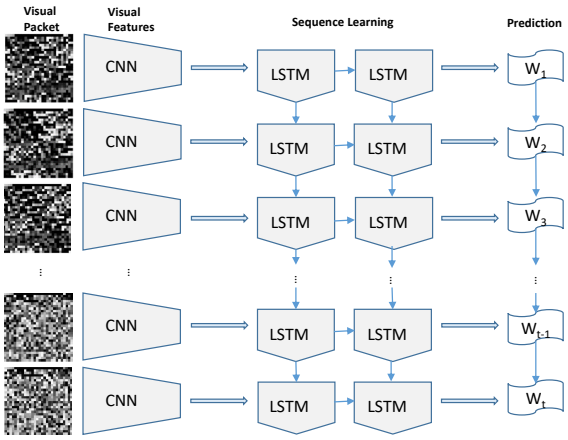

Fig. 11. Long-term Recurrent Convolutional Networks for traffic classification

B. VAE-GAN

Traditional deep learning methods used in NIDS suffer from the problem that the False Detection Rate (FDR) is somewhat high due to the inherent drawback of anomaly detection and the unbalance training data which have much more normal traffic data than abnormal traffic data. The unbalanced category distribution will affect the deep learning model much, specially it will increase false positive rate.

VAE-GAN is a new model combines VAE(III-C) and GAN(III-F), the discriminator learn to discriminate the real(x), reconstructed($G(E(x))$) and generated($G(z)$) data.

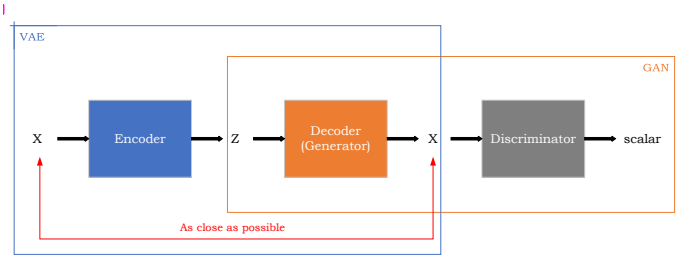

Fig. 12. VAE-GAN

We can use the decoder (generator) as shown in Fig. 10 in VAE-GAN to upsample the minority class of attacking data in datasets instead of using classical VAE, which is more reasonable than Abdulhammed [12] and Mirsky [13] did. Furthermore, there is another profit brought by the well-trained discriminator. After the false detection of anomaly detection (if there are), the discriminator can discriminate the normal packets that classified as intrusion .

V. Conclusion

Protection for a cyber-security system usually lags behind the attacks so intrusion detection plays a significant role in making up the time interval before the reaction of the security system. The real-time prosperity and scalability are really important. Combining with advances in ML algorithm development offers a rich opportunity to apply neural network-based DL approaches to cyber-security applications to detect new variants of malware and zero-day attacks. Then we aims at keep the networks model time-saving and effective in networks node where lacks computation resources. We give an outline of recent related works in intrusion detection utilized deep learning ways. Different with other surveys or papers, we explore the overview of development by specifically listing the influential works with creative thesis instead of all related works. In future works, the cascading connection of malicious activities throughout an attack lifecycle as we mentioned can be explored to contribute to build an intelligent and efficient networks intrusion detection system. Besides, an idealized framework based on VAE-GAN is designed for processing intrusion detection.

References

[1] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2. IEEE, 2000, pp. 12–26.

[2] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer networks*, vol. 34, no. 4, pp. 579–595, 2000.

[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*. IEEE, 2009, pp. 1–6.

[4] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[5] N. Gao, L. Gao, Q. Gao, and H. Wang, "An intrusion detection model based on deep belief networks," in *2014 Second International Conference on Advanced Cloud and Big Data*. IEEE, 2014, pp. 247–252.

[6] K. K. Nguyen, D. T. Hoang, D. Niyato, P. Wang, D. Nguyen, and E. Dutkiewicz, "Cyberattack detection in mobile cloud computing: A deep learning approach," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2018, pp. 1–6.

[7] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2016, pp. 195–200.

[8] M. Z. Alom, V. Bontupalli, and T. M. Taha, "Intrusion detection using deep belief networks," in *2015 National Aerospace and Electronics Conference (NAECON)*. IEEE, 2015, pp. 339–344.

[9] Y. Li, R. Ma, and R. Jiao, "A hybrid malicious code detection method based on deep learning," *International Journal of Security and Its Applications*, vol. 9, no. 5, pp. 205–216, 2015.

[10] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-based feature learning for cyber security applications," in *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 3854–3861.

[11] M. Z. Alom and T. M. Taha, "Network intrusion detection for cyber security using unsupervised deep learning approaches," in *2017 IEEE National Aerospace and Electronics Conference (NAECON)*. IEEE, 2017, pp. 63–69.

[12] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE sensors letters*, vol. 3, no. 1, pp. 1–4, 2019.

[13] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[14] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*. IEEE, 2017, pp. 712–717.

[15] Y. LeCun, L. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard *et al.*, "Learning algorithms for classification: A comparison on handwritten digit recognition," *Neural networks: the statistical mechanics perspective*, vol. 261, p. 276, 1995.

[16] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[19] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[20] R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," *South African Computer Journal*, vol. 56, no. 1, pp. 136–154, 2015.

[21] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International Conference on Information Processing in Medical Imaging*. Springer, 2017, pp. 146–157.

[22] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient gan-based anomaly detection," *arXiv preprint arXiv:1802.06222*, 2018.

[23] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016.

[24] H. Chen and L. Jiang, "Gan-based method for cyber-intrusion detection," *arXiv preprint arXiv:1904.02426*, 2019.

[25] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in

Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 2625–2634.