

An Efficient Cyber Security Intrusion Detection Intrusion Detection using Deep Learning Technique

¹Yalla Venkatesh, ²P Rama Krishna

^{1,2}Dept. of Computer Science & Engineering, KIET, Kakinada, AP, India

Abstract

Intrusion Detection Systems are core part of cyber security measures in all organizations. With increasing amount of data available online in digitized form, this has resulted in an ever growing need for stringent cyber security measures against data breaches and malware attacks. Rising number of attacks coupled with new variants of malware being released on a frequent basis require automated intrusion detection systems. With the state of the art performance of the Deep Learning based Models in the field of computer vision, natural language processing and speech recognition and Deep learning techniques are now being applied to the field of cyber security. Deep Learning has been efficiently implemented in Intrusion Detection Systems.

Keywords

Deep Learning; Intrusion Detection Systems; Anomaly Based Detection; IDS

I. Introduction

The Role of cyber security in today's information age is critical. Increased number of attacks and evolving nature of the malware require timely intervention. The large scale availability of data online makes it vulnerable to attacks such as malware attacks, phishing attacks, Spoofing, Denial of Service attacks and injection attacks. The feasible requirement is to deploy automated intrusion detection systems (IDS) based on advanced machine learning algorithms to automatically detect attacks and classify them. Deep Learning based algorithms are providing better results than existing machine learning solutions in this scenario. The objective of the paper is to provide a review of existing Intrusion Detection Systems which are implemented using Deep Learning Algorithms. The paper also categories the existing methods based on the different Deep Architectures and provides the reader with salient features of each of the methods together with the type of detection used to identify the intrusion. It also provides a classification of the Deep Learning Models and background information to arm the reader before exposing him/her to the review details.

The paper is organised as follows: section2 describes different Intrusion Detection Systems with respect the location of the IDS and the nature of the detection, section3 introduces various Deep Learning techniques segregating them into four types of deep learning models. Section 4 and section 5 contain details of the performance metrics and datasets applicable in Intrusion detection Systems. Section 6 contains the summary analysis of the methods examined and key findings that stem from it. We draw some concluding remarks in Section 7.

II. Intrusion Detection Systems

Intrusion Detection can be defined as "the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource"[1]. The goal of the intrusion detection systems is to identify entities that undermine the security controls of a system. The role of the IDS is always passive since it is

concerned with gathering, identifying, logging and alerting. Intrusion Detection Systems can be classified either based on its location in the network or the type of detection that is used. Based on the location of the IDS, it can be differentiated as either Network based or Host Based [2]. On the basis of the nature of the detection used, the IDS can be classified as Signature based or Anomaly Based. Network based intrusion detection can identify unauthorized, illicit, and anomalous behaviour using the nature of network traffic. A network IDS, using either a network tap, span port, or hub collects packets that traverse a given network. With the captured data, the system then identifies and flags any suspicious traffic. Unlike an intrusion prevention system, an intrusion detection system does not actively block network traffic. Often referred to as HIDS, host based intrusion detection is able to identify unauthorized, illicit, and anomalous behaviour on a specific device.

This is achieved using an agent installed on each system, monitoring and alerting on local OS and application activity. The installed agent then uses a combination of signatures, rules, and heuristics to identify unauthorized activity.

IDS can deploy signature based detection, using known traffic data as a reference to analyse potentially unwanted traffic. This type of detection is very fast and easy to configure. However, an attacker can slightly modify an attack to render it undetectable by a signature based IDS. Still, signature-based detection, although limited in its detection capability, can be accurate to a good degree.

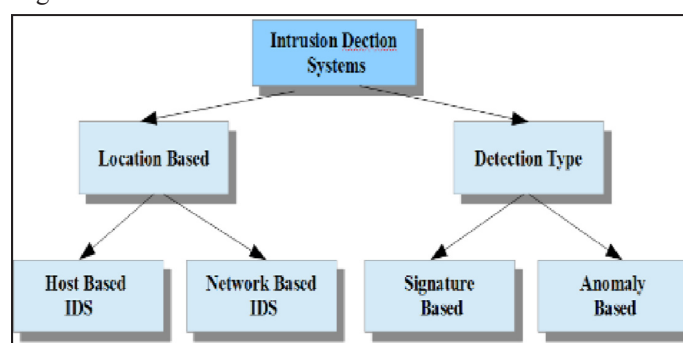


Fig.1. Categories of Intrusion Detection Systems

An IDS that looks at network traffic and is able to distinguish between normal traffic and invalid or incorrect traffic. This method is useful for detecting unwanted traffic not previously seen. It can also detect malformed packets. For the system to detect the anomalous behaviour, it must first be trained to recognize normal system activity. The two phases of anomaly detection systems consist of the training phase and the testing phase. The training phase consists of building a profile of normal behaviours and testing phase has the existing traffic being compared with the profile created in the training phase. Artificial Intelligence based techniques have proved useful in detecting such behaviours especially neural networks.

III. Deep Learning Architectures

Deep learning architectures have gained traction currently due to their widespread success in the field of computer vision. State of the art results have also been obtained in the field of Speech Recognition and in Natural Language processing. Deep Learning Architectures are comprised of multiple levels of non-linear operations similar to neural networks with many hidden layers. The success of the Deep learning architectures lies in using fast learning algorithms and suitable hardware for fast and efficient solutions. The development of GPU accelerated computing has led to the increase in their development and lead to faster convergence of the algorithms. They provide an increased level of abstraction and can be distinguished by the following capabilities:

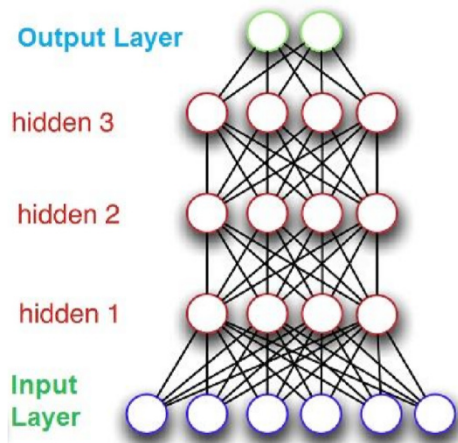


Fig. 2: A Deep Learning Network with three hidden layers

- Increased depth characterised by the number of the node layers (hidden layers) through which the data passes in a multi step process of classification.
- Capable of handling complex features through its Feature Hierarchy.
- Handling of large, high dimensional data sets with large number of parameters in a non-linear way.
- Discover latent structures within unlabelled unstructured data also known as Automatic Feature Learning/extraction.

Fig. 2 show a Deep Learning Network with 3 Hidden Layers. Deep Networks are stacked neural networks. Layers are made of nodes. A node combines input from the data with a set of coefficients, or weights that either amplify or dampen that input, thereby assigning significance to inputs for the task the algorithm is trying to learn. Each of the Node layers train on a distinct set of features based on the output from the previous layer. Hence the learning is more effective if the number of layers is large. For the purpose of the review, based on their architecture the Deep learning networks can be classified into the following broad categories: Convolutional Neural Networks, Deep Generative Models, Autoencoder Based and RNN Based Models.

A. CNN Based Models

Convolutional Neural Networks are designed to process inputs which are image based [3]. The first successful applications of Convolutional Networks were developed in the 1990's by YannLeCun. The layers of a CNN have neurons arranged in 3 dimensions: width, height and depth. The CNN uses 3 layers of architecture: Convolutional Layer, Pooling Layer, and Fully Connected Layer. The layers are stacked to form a full CNN architecture. In the convolutional layers, a CNN utilises various

kernels to convolve the whole image as well as the intermediate feature maps, generating numerous feature maps in the process. Apart from the convolutional layers, they also often feature pooling layers. Pooling is a way to filter out details: a commonly found pooling technique is max pooling, where we take say 2×2 pixels and pass on the pixel with the most amount of the required detail. Usage of CNN is not limited to image data. These non-image applications of CNN use a Feed forward neural network to the end to further process the data, which allows for highly non-linear abstractions. Fig.3 shows a convolutional neural network.

1. Convolutional Layer

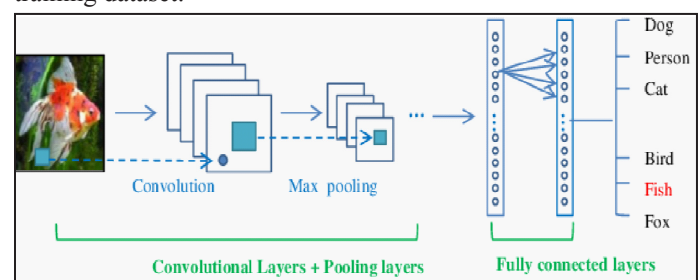
The convolutional layer constitutes the core building block of a CNN. The layer's parameters consist of a set of filters (or kernels). The Kernels have a small receptive field, but extend through the full depth of the input volume. During forward pass, each filter is applied to the width and height of the input volume. This results in a 2-dimensional activation map of that filter. As a result, the network learns filters which activate when they encounter a specific type of feature at some spatial position in the input. After every convolution, an operation called ReLU (Rectified Linear Units) is used to introduce non-linearity which results in a rectified Feature Map.

2. Pooling Layers

Generally, a pooling layer or sub sampling follows a convolutional layer, and can be used to reduce the dimensions of feature maps and network parameters. Depending on the Pooling method used (Max, Average, Sum) an element is chosen from each window. The window is determined using a spatial neighbourhood. Similar to convolutional layers, pooling layers are translation invariant, because their computations take neighbouring pixels into account.

3. Fully Connected Layers

The purpose of the Fully Connected layer is to use the high level features obtained from convolutional and pooling layers for classifying the input image into various classes based on the training dataset.



CNN have been the focus of large scale image and video recognition. Image-Net Large-Scale Visual Recognition Challenge (ILSVRC) has played a critical role in the impetus for deep visual recognition architectures.

B. Deep Generative Models

Deep Generative Models are useful for modelling of generative models of images and natural data. The multiple layers of stochastic units make the inference and learning challenging. Popular generative models are Deep Belief Networks (DBN) and Deep Boltzmann machines (DBM).

In their seminal work in [4], Hinton describes how to train deep belief networks with multiple hidden units. A DBN is a graphical model with undirected connections at the top hidden layers and

with directed connections in the lower layers. Deep Boltzmann Machines proposed by Salakhutdinov et al.[5] are composed of many Restricted Boltzmann Machines. These networks are “restricted” to a single visible layer and single hidden layer. The connections are formed between the layers with the restriction that no two units in a layer may be connected. This forms the Restriction of the RBM. The hidden units are trained to capture higher-order data correlations that are observed at the visible units. Initially, apart from the top two layers, which form an associative memory, the layers of DBN are connected only by directed topdown generative weights. RBMs are an attractive option as building blocks for DNN due to their ease of learning these connection weights. The initial pre-training proceeds in an unsupervised greedy layer-by-layer manner, called as contrastive divergence [6]. Once the RBM learns the structure of the input data as it relates to the activations of the first hidden layer, then the data is passed one layer down the next. The first hidden layer takes on the role of visible layer. The activations become the input for the next layer, and they are multiplied by weights at the nodes of the second hidden layer, to produce another set of activations.

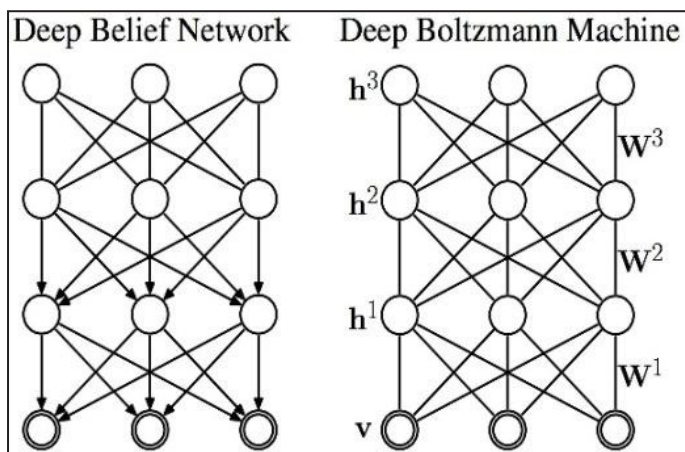


Fig. 4: Deep Generative Models

This method of creating sequential sets of activations by grouping features and creating further groups of features is termed feature hierarchy, using which neural networks learn more complex and abstract representations of data. With each new hidden layer, the weights are adjusted until that layer is able to approximate the input from the previous layer. This provides a greedy, layer-wise and unsupervised pre-training. It requires no labels to improve the weights of the network allowing the training to occur on unlabeled data. DBMs have undirected connections in all layers. Deep energy models or DEMs be seen as having deterministic hidden units for the lower layers and stochastic hidden units at the top hidden layer.

C. Autoencoder Based Models

An autoencoder applies back propagation on a neural network, setting the target values to be same as the inputs. They belong to the class of unsupervised learning algorithms. The autoencoder takes an input vector and maps it to a hidden representation using a process called encoding. The Hidden Unit called Code is then mapped back to the input space using a decoding process. The autoencoder is designed to minimise the Reconstruction error which is represented by a distance between the input and the output. When the encoder has a non-linear form or is multilayered then the autoencoder learns useful representations of the data if there is a limit on the number of hidden units [7]. The autoencoder

can still be made to discover interesting structure in the data, for a large number of hidden units less than the input units. These autoencoders impose a sparsity constraint on the hidden units. Such autoencoders are termed Sparse. Denoising autoencoders add noise to the data and learn about that data by attempting to reconstruct it. The network then attempts to recognize the features within the noise that will allow it to classify the input and help understand it better. It uses a denoising criterion in unsupervised training to learn higher level representations of the input data. During the training process, the distance between that model and the benchmark through a loss function is measure using a loss function. It minimizes the loss function by resampling the shuffled inputs and re-reconstructing the data. This process is repeated until it finds those inputs which bring its model closest to what it has been told is true.

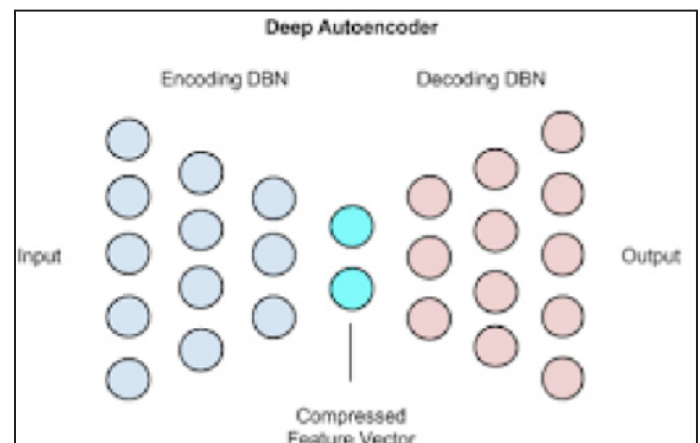


Fig. 5: Autoencoder – encoding and decoding data.

D. Recurrent Neural Network Based Models

The basic feature of a Recurrent Neural Network (RNN) is the use of feed-back connections in the network. This causes the activations to flow around in a loop. That enables the networks to do temporal processing and hence learn sequences, e.g., perform time series analysis or temporal association/prediction. RNNs are capable of processing an input sequence one element at a time, maintaining in their hidden units a „state vector“ that implicitly contains information about the history of all the past elements of the sequence. When we consider the outputs of the hidden units at various discrete time steps as if they were the outputs of different neurons in a deep multilayer network, it becomes evident as to how backpropagation can be used for training RNN [8].

Recurrent neural network architectures have many different forms. One common type consists of a standard MultiLayer Perceptron (MLP) which has added loops. These exploit the powerful non-linear mapping capabilities of the MLP, and also have some form of memory. Learning can be effected by using gradient descent procedures similar to those using the back-propagation algorithm for feed-forward networks in simple networks using deterministic activation functions. When the activations are stochastic, simulated annealing approaches may be suitable. LSTM networks are a special type of RNN first introduced by Hochreiter&Schmidhuber in 1997[9]. The Long Short-Term Memory network, or LSTM network, is a recurrent neural network that is trained using Back propagation algorithm.

The vanishing gradient problem seen in the RNN can be solved by using the LSTM. LSTM networks are mostly used to address difficult sequence problems in machine learning and achieve

state-of-the-art results. Instead of neurons, LSTM networks have memory blocks that are connected through layers. It contains gates that manage the block's state and output. Given an input sequence, each gate within a block uses the sigmoid activation units to control whether they are triggered or not, making the change of state and addition of information flowing through the block conditional.

There are three types of gates within a unit:

- Input Gate: decides values to be used from the input to update the memory state.
- Output Gate: decides the output based on input and the memory of the block.
- Forget Gate: decides the information that must be removed from the block.

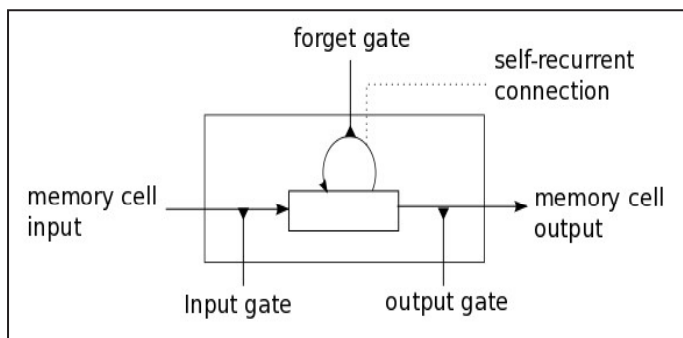


Fig. 6: LSTM cell with 3 types of gates

Each unit is like a mini-state machine where the gates of the units have weights that are learned during the training procedure. In this way, it is possible to achieve sophisticated learning and memory from a layer of LSTMs, and create higher-order abstractions from multiple such layers.

IV. Performance Metrics For Intrusion Detection Systems

For determining the accuracy of IDS, there are four possible states for each activity observed. A true positive state is a situation when the IDS identify an activity as an attack and the activity is actually an attack. This state is termed as a identification of an attack. A true negative state is also similar. This is when the IDS identify an activity as an acceptable behaviour and the activity is actually acceptable. A true negative can be termed as successfully ignoring acceptable behaviour. Both the above states are not damaging to the IDS. A false positive is when the IDS identify an activity as an attack but the activity is acceptable behaviour. A false positive is therefore a false alarm. A false negative state is a critical and dangerous state. This is when the IDS identify an activity as acceptable when an attack is taking place. Hence, the IDS ignore the attack.

Generally, Detection Rate (DR) and False Alarm Rate (FAR) are used as the metrics of IDS evaluation. The DR shown in formula (1) signifies a ratio of intrusion instances detected by IDS model. The FAR referenced by formula (2) is a ratio of misclassified normal instances. Based on a confusion matrix, equations of the metrics are as follow (TP: true positive, TN: true negative, FP: false positive, FN: false negative).

$$DR = \frac{TP}{TP+FN} \quad (1)$$

$$FAR = \frac{FP}{FP+TN} \quad (2)$$

In other studies in measuring the performance of the Intrusion detection systems, we can observe that accuracy(3), error rate (4), recall (5) and specificity(6) are used to evaluate the performance of the detection models. The accuracy rate shows the overall correct detection accuracy of the dataset, ER refers to the robustness of the classifier, recall indicates the degree of correctly detected attack types of all cases classified as attacks, and specificity shows the percentage of correctly classified normal data. In the above, higher accuracy and recall with a lower ER indicate good performance. The formulas of the above criteria are calculated as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

$$Error\ Rate = \frac{FP+FN}{TP+TN+FP+FN} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$Specificity = \frac{TN}{TN+FP} \quad (6)$$

V. Datasets For Intrusion Detection

Most commonly used dataset is the KDD99 dataset. The KDD99 dataset has been obtained in 1999 from the DARPA98 network traffic dataset. It was the benchmark dataset used for the international Knowledge Discovery and Data Mining Tools Competition, and also the most popular dataset that has ever been used in the intrusion detection field. Each TCP connection has 41 features with a label which specifies the status of a connection as either being normal, or a specific attack type. There are 38 numeric features and 3 symbolic features, falling into the following four categories:

- Basic features: 9 basic features were used to describe each individual TCP connection.
- Content features: 13 domain knowledge related features were used to indicate suspicious behaviour having no sequential patterns in the network traffic.
- Time-based traffic features: 9 features were used to summarize the connections in the past 2s that had the same destination host or the same service as the current connection.
- Host-based traffic features: 10 features were constructed using a window of 100 connections to the same host instead of a time window, because slow scan attacks may occupy a much larger time interval than 2 s.

VI. Conclusion

The paper has examined the various Deep Learning Models to aid the detection of malware and unwanted traffic. It classifies the method used to identify the intrusion in each of the cases. It can be seen from the classification that Autoencoders and Recurrent Neural Networks outperform the CNN based models with high accuracy percentages. This is logical since CNN are basically designed for image processing applications. Hence more ensemble models utilizing the Autoencoder and the RNN based methods can be incorporated in models for improved accuracy.

The field of cyber security will continue to see improvements due to the Deep Learning Algorithms leading to more secure and more adaptable systems. The future looks promising for Intrusion Detection Systems as the technology growth leads to much faster processing hardware and design of more efficient deep learning algorithms.

References

- [1] <https://www.sans.org/security-resources/idfaq/what-is-intrusion-detection/1/1>.
- [2] https://www.owasp.org/index.php/Intrusion_Detection
- [3] <http://egrcc.github.io/docs/dl/deeplearningbook-convnets.pdf>. MIT Deep Learning Book.
- [4] Geoffrey E. Hinton, Simon Osindero and Yee-Whye, A fast learning algorithm for deep belief nets, Neural Computation. 2006.
- [5] R. Salakhutdinov, G.E. Hinton, "Deep boltzmann machines", in: Proceedings of the AISTATS. 2009
- [6] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," Neural Computation., vol. 14, pp. 1771–1800. 2002
- [7] Quoc V. Le, "A Tutorial on Deep Learning Part 2: Autoencoders, Convolutional Neural Networks and Recurrent Neural Networks".
- [8] Alex Graves and Navdeep Jaitly, Google DeepMind, UK, "Towards End-to-End Speech Recognition with Recurrent Neural Networks" [9] Sepp Hochreiter, Jürgen Schmidhuber, "Long Short Term Memory", Neural Computation, Pages 1735-1780, Volume 9 Issue 8, November 15, 1997.
- [10] Daniel Gibert, "Convolutional Neural Networks for Malware Classification", Thesis for Masters in Artificial Intelligence, 2016.
- [11] Bojan Kolosnjaji, Apostolis Zarras, George Webster, and Claudia Eckert. "Deep Learning for Classification of Malware System Call Sequences". In 29th Australasian Joint Conference on Artificial Intelligence (AI), 2016.
- [12] Shun Tobiyama, Yukiko Yamaguchi, Hajime Shimada, Tomonori Ikuse and Takeshi Yagi. "Malware Detection with Deep Neural Network Using Process Behavior". In IEEE 40th Annual Computer Software and Applications Conference, 2016.
- [13] George E. Dahl, Jack W. Stokes, Li Deng and Dong Yu. "Large-scale malware classification using Random Projections and Neural Networks". In ICASSP 2013.
- [14] Zhenlong Yuan, Yongqiang Lu, Zhaoguo Wang and Yibo Xue. "Droid-Sec: deep learning in android malware detection", Proceedings of the 2014 ACM conference on SIGCOMM, Chicago, Illinois, USA, August 17-22, 2014.
- [15] Joshua Saxe and Konstantin Berlin. "Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features". In arXiv:1508.03096v2. arXiv preprint (2015).