SSH-Brute Force Attack Detection Model based on Deep Learning

Stephen Kahara Wanjau,
School of Computing and Information Technology,
Murang'a University of Technology,
Murang'a, Kenya.

Geoffrey Mariga Wambugu,
School of Computing and Information Technology,
Murang'a University of Technology,
Murang'a, Kenya.

Gabriel Ndung'u Kamau,
School of Computing and Information Technology,
Murang'a University of Technology,
Murang'a, Kenya.

Abstract: The rising number of malicious threats on computer networks and Internet services owing to a large number of attacks makes the network security be at incessant risk. One of the predominant network attacks that poses distressing threats to networks security are the brute force attacks. A brute force attack uses a trial and error algorithm to decode encrypted data such as passwords or Data Encryption Standard keys, through exhaustive effort (using brute force) rather than using intellectual strategies. Brute force attacks resemble legitimate network traffic, making it difficult to defend an organization that rely mainly on perimeter-based security solutions a major challenge. For stopping the occurrence of such attacks, several curable steps must be taken. This paper proposes an efficient mechanism for SSH-Brute force network attacks detection based on a supervised deep learning algorithm, Convolutional Neural Network. The model performance was compared with experimental results from 5 classical machine learning algorithms including Naive Bayes, Logistic Regression, Decision Tree, k-Nearest Neighbour, and Support Vector Machine. Four standard metrics namely, Accuracy, Precision, Recall, and the F-measure were used. Results show that the CNN-based model is superior to the traditional machine learning methods with 94.3% accuracy, a precision rate of 92.5%, recall rate of 97.8% and F1-score of 91.8% in terms of the ability to detect SSH-Brute force attacks..

Keywords: Convolutional Neural Network, Deep Learning, Feature Selection, Network Security, Occam's razor principle, SSH Brute force

1. INTRODUCTION

Computer network attackers have acquire advanced skills and are exploiting unknown vulnerabilities to bypass security solutions. Among the leading network attacks are the brute force attacks [7]. Brute force attacks are becoming harder to successfully detect on a network level due to the growing ubiquity of high-speed networks and increasing volume and encryption of network traffic [25]. A brute force attacking application proceeds through all possible combinations of legal characters in sequence until they find the correct input. The longer the password, the more time it will typically take to find the correct input. Most common brute force attacks use a password dictionary that contains millions of words to test. Successful brute force attacks not only give hackers access to data, applications, and resources, but can also serve as an entry point for further attacks.

Several signs can be construed to be indicators of a brute force attack. Among them include, several failed login attempts from the same IP address; logins with multiple username attempts from the same IP address; logins for a single account from many different IP addresses; failed login attempts from alphabetically sequential usernames and passwords; logins with a referring URL of someone's mail or IRC client; excessive bandwidth consumption over the course of a single session and a large number of authentication failures [24].

Secure Shell (SSH) is one of the most popular communication protocols on the Internet widely used by developers, webmasters, and system administrators. SSH permits one to gain remote access to a new cloud service or a dedicated box in just seconds using an encrypted communication channel. SSH-Brute force attacks tries to gain access to a remote machine by performing authentication attempts, systematically checking all the possible passwords until the correct one is found [26] on the Secure Shell protocol. Normally, attackers may use applications and scripts as brute force tools. These tools try out numerous password combinations to bypass authentication processes. If the host is exposed directly to the Internet or Wide Area Network (WAN) and the SSH service is running on the host, it becomes a subject of constant brute force attacks performed by automated scripts such as hydra. In other cases, attackers try to access web-based applications by searching for the right session ID. Normally, human-selected passwords are characteristically weak as users tend to choose simple passwords which are easier to remember. Sometimes, they don't change the machine's default password or simply use the user name as the password. This makes such machines prone to successful brute force attacks.

In the 2018 report by Verizon [38], brute force attacks were ranked top among attack types detected by IPS (pg.51). Microsoft Office 365 was also a target of massive brute force attacks [32]. The attackers tried logging in with different versions of employees' Office 365 usernames, suggesting they may already possess some combination of employee names and passwords and were seeking valid Office 365 usernames for data access or spear phishing campaigns. The password data could have been obtained in a database breach of a service like Yahoo or a phishing attack, given that password reuse across accounts remains rampant. Alibaba, one of the world's largest retailer and e-commerce Company suffered a massive brute-force attack on its e-commerce site, TaoBao. Using a database of about 99 million usernames and passwords, the attackers managed to compromise approximately 21 million accounts [31].

GitHub, a source code management system, fell victim to massive brute force attacks in 2013, which successfully compromised some accounts emanating from about 40,000 unique IP addresses [3]. Similarly, on the same year, WordPress, one of the most high-profile open source content management system in use today was a target of massive brute force attacks. A large botnet with more than 90,000 servers attempted to log in by cycling through different usernames and passwords [17]. In 2016, a British national pleaded guilty in German court to launching an automated botnet brute-force attack designed to infect 1.25 million German routers with Mirai malware and causing €2 million ($2.33 million) in damage [30].

In general, to prevent against brute force attacks, network administrators can employ several measures that include (i) adding to password complexity, thereby making any process of guessing a password take significantly longer [28]. For example, some websites will require passwords of 8-16 characters, with at least one letter and number with special characters (such as “$”), as well as not allowing a user to have their name, username or ID in their password, (ii) login attempts that focus on predefined time and number of attempts a user will make to input passwords/usernames [19], (iii) use of Captchas that show a box with warped text and asks the user what the text in the box is. This prevents bots from executing automated scripts that appear in brute force attacks, and (iv) two-factor authentication (a type of multi-factor authentication) that adds a layer of security to the primary form of authentication. Usually, two-factor security requires two forms of authentication (for example, to sign in to a new Apple device, users need to put in their Apple ID along with a six-digit code that is displayed on another one of their devices previously marked as trusted).

Existing tools and methods to detect and avoid these attacks have mainly remained static over the years and are built on common data models such SSH tunneling (also called SSH port forwarding), firewalls and common pairs of username and password [5]. Thus, an efficient mechanism for SSH-Brute force network attacks detection is required in order to organically grow with the ever-expanding attack structures of today’s cyber environment.

In recent years, machine-learning techniques have been actively employed to network security and are becoming a prevalent way of detecting advanced attacks with unexpected patterns [26], [29], [37]. More recently, deep neural networks have been applied in studies involving network security [6], [16] since they have a powerful mechanism for supervised learning. They can represent functions of increasing complexity, by including more layers and more units per layer in a neural network [9]. In network intrusion detection, deep neural networks can be used to discover patterns of malicious and benign traffic hidden within huge amounts of structured data [6]. However, its efficacy in the context of SSH-Brute force attacks detection has not been systematically investigated despite its tremendous success in other application domains such as malware detection and spam mail detection. This paper proposes an efficient mechanism for SSH-Brute force network attack detection based on a supervised deep learning algorithm.

2. RELATED WORKS

Several studies have investigated detection of SSH brute force attacks. The study by [14] proposed an approach of detecting attacks in individually sly activities, which operates in unsuspected manner in a SSH Brute-Force attack. The study depended on two elements; Site Aggregates Analyser (to observe the activities and attacks which occur in the sites and detect it) and Attack Participants Classifier (to analyze and classify the attack's participant). Another study [41] proposed a protocol called Password Guessing Resistant Protocol (PGRP), derived upon revisiting proposals previously designed to avoid such attacks. The system was divided into three parts namely: User & Password Authentication, IP Authentication, and Cookie Authentication.

The study conducted by [1] examined brute force attacks based on SSH log files to discover unsuccessful logins and then establish if these unsuccessful authorized IP’s belong to attackers or to trusted users. The study suggested a technique, Detecting Distributed Brute Force Attack on a Single Target (DBFST), a strategy to determine who should transact with the IP addresses based on the IP kind, and prevent the attackers IP's from attempting to login to the system. The technique was able to detect the attacks from the same subnet or network, and block the attackers’ networks.

In another study by [26], the researchers aggregated network flow data along with a machine learning approach for the detection of SSH brute force attacks at the network level. Classification algorithms (k-Nearest Neighbor, decision trees, artificial neural network and Naïve Bayes) were used to build models for extracting discriminative features for the detection of brute force attacks, collecting real SSH traffic from a campus network. The dataset also contained data similar to attack network traffic (failed login data produced by legitimate users that have forgotten their passwords). They analyzed brute force versus normal SSH traffic in order to define the features that can be discriminative enough to discriminate between normal and attack traffic.

In a recent study, [6] used deep learning for both supervised network intrusion detection and unsupervised network anomaly detection. In their study, a feedforward fully connected Deep Neural Network (DNN) was used to train a network intrusion detection system through supervised learning. An auto encoder was also used to detect and classify attack traffic via unsupervised learning in the absence of labelled malicious traffic. The study evaluated the models using two recent network intrusion detection datasets with known ground truth of malicious versus benign traffic, the CIC IDS 2017 dataset [33] and ISCX IDS 2012 [34] dataset. The study results demonstrated that the DNN outperform other machine learning based intrusion detection systems.

In their study, [16] utilized deep-learning techniques to develop a convolutional neural network (CNN) model to detect network intrusions. The study used CIC IDS 2018 dataset where the numerical data was converted into images for training. The CIC-2018 dataset consist of 10 days of sub-datasets collected on different days through injecting 16 types of attacks generated using CICFlowMeter-V3 [2] containing about 80 types of features. The model performance was evaluated by comparing experimental results with that of a recurrent neural network (RNN) model.

With widespread adoption of cloud computing, coupled with extensive deployment of plenty of Web applications, the need for anomaly detection from the packet payloads is becoming a challenge. Researchers, [22] proposed a feature engineering method that constructs block-based features of the packet payload to adaptively detect anomalies through block sequence extraction and block embedding. In addition, a deep neural network was designed to learn the representation of the packet payload based on Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN) and evaluated the framework using three public dataset namely, CSIC 2010 HTTP dataset [8] which contains generated traffic targeted to an e-commerce Web application, ISCX 2012 dataset that contains network traffic which aims to describe network behaviours and intrusion patterns, and the CIC IDS 2017 dataset that contains both normal traffic and up-to-date attacks which resemble the true data.

Our study builds upon the works by [6] and [16]. The study by [6] showed that deep neural networks can outperform other machine learning based intrusion detection systems, while being robust in the presence of dynamic IP addresses. The study by [16] employed deep-learning techniques and developed a convolutional neural network (CNN) model for network anomaly detection using CSE-CIC-IDS 2018 dataset. Their model performance was compared with a recurrent neural network (RNN) model. The CNN model performance was higher than that of the RNN Model.

This work propose to extend these two studies by proposing an SSH-Brute force attack detection model based on a supervised deep learning algorithm, Convolutional Neural Network. The choice of CNN is because the algorithm can cope with tabular data that contains categorical variables of high cardinality, which are exhibited by the dataset used [6]. The study trained the images based on the proposed model and evaluate its performance by comparing experimental results with that of 5 classical machine learning algorithms namely Naive Bayes, Logistic Regression, Decision Tree, k-Nearest Neighbor, and Support Vector Machine.

3. PROPOSED APPROACH

The study adopted the design science research methodology. The study proposed a method that comprises of feature selection and a deep learning algorithm. Feature selection extract the most relevant features or attributes to identify the instance to a particular group or class. The deep learning algorithm builds the necessary intelligence or knowledge using the results obtained from the feature selection [35] using a dataset. Figure 1 shows the proposed deep learning classifier for SSH-Brute force attack detection.

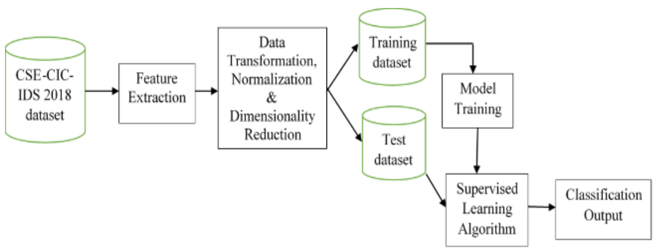

Figure 1: Proposed Deep Learning Classifier for SSH-Brute force attack detection.

3.1 Dataset

This study used, the CIC- IDS 2018 benchmark dataset [16] that includes the contemporary activities of benign and malicious attacks which depicts the real-time network traffic. In this dataset, benign background traffic was collected using B-profile system and contains the characteristics of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols. The network traffic was collected for five days and dumped with normal activity traffic on one day, and attacks injected on other days. The various injected attacks include Brute Force FTP, Brute Force SSH, Denial of Service, Heartbleed, Web Attack, Infiltration, Botnet and Distributed Denial of Service [39].

The dataset was generated using CICFlowMeter-V3 [2] and contains 83 types of features that provide forward and backward directions of network flow and packets, including one column for the label, and another Column for the FlowID. Five of the features are categorical including, ‘SourceIP’, ‘DestinationIP’, ‘SourcePort’, ‘DestinationPort’ and ‘Protocol’.

The remaining 78 features are continuous. This pre-processed network flow data is provided as CSV files that can be fed into the machine learning pipeline. The dataset not only contain modern-day attacks, but is also created in a manner that follow established guidelines of reliable intrusion detection datasets in terms of realism, evaluation capabilities, total capture, completeness, and malicious activity [34]. The dataset contains multinomial class labels for each of the attack types carried out in the flow records. Table 1 shows the dataset used in this work.

Table 1. Type of attacks and amounts of subdata.

Sub-Data	Type of attacks	Total samples
Sub-Data 1	Benign	1,048,574
	DoS Hulk	
	DoS-SlowHTTPTest	
Sub-Data 2	Benign	1,044,751
	Brute Force-FTP	
	Brute Force-SSH	
Sub-Data 3	Benign	1,040,548
	DoS-GoldenEye	
	DoS-Slowloris	
Sub-Data 4	Benign	7,889,295
	DDoS-LOIC-HTTP	
Sub-Data 5	Benign	1,048,575
	DDoS-HOIC	
	DDoS-LOIC-UDP	
Sub-Data 6	Benign	1,042,965
	Brute Force -Web	
	Brute Force -XSS	
	SQL Injection	
Sub-Data 7	Benign	1,042,867
	Brute Force -Web	
	Brute Force -XSS	
	SQL Injection	
Sub-Data 8	Benign	606,902
	Infiltration	
Sub-Data 9	Benign	328,181
	Infiltration	
Sub-Data 9	Benign	1,044,525
	Bot	

Before the data can be used, it requires further pre-processing. The dataset pre-processing steps used are as follows.

3.2 Feature Selection

Feature selection is the process of determining the features to be used for learning by removing those that are not relevant or are redundant [13]. The main goal is to avoid overfitting the data in order to make further analysis possible. Feature selection techniques do not alter the original representation of the data. The subset of features selected followed the Occam's razor principle and also give the best performance according to some objective function. The pre-processed network flow data has 83 columns (e.g., duration, number of packets, number of bytes, and length of packets) that can be used as features, plus one label column and one flow ID column.

Since seven different kinds of attacks are contained in this dataset (i.e., brute-force against the SSH and Web, Heartbleed, botnet, denial of service (DoS), distributed denial of service (DDoS), cross-site scripting (XSS) and SQL injection attacks against websites, and infiltration) [6], a feature subset need to be selected for this study. A filter algorithm was used for subset selection. Filters work without taking the classifier into consideration making them very computationally efficient. The Markov Blanket Filtering (multivariate) method was used which finds features that are independent of the class label so that removing them will not affect the accuracy. According to [12], "a good feature subset is one that contains features highly correlated with the class yet uncorrelated with each other." Since the focus of the study was on detecting SSH-Brute force attacks, this filtering method was considered appropriate in identifying a subset data for the study.

3.3 Data Transformation

In order to develop a CNN-based SSH-Brute-force attack detection model, converting the selected features from the dataset into images was required. For computers, the essence of images is the array of pixel values. In this study, each of the pre-processed record of the dataset was transformed to generate a two-dimensional image matrix group that meets the requirements. Each labelled data was converted into 13x6 size of images since each data contains 78 features except the 'Label' feature. The 'Label' was used for image classification. Given the pixel values of the image range from 0 to 255, the attribute of each record is given as:

$$P_i = \mathrm{r}_i \text{ x } 255$$

Where, P_i is the element of the array and the images produced are inputted into the CNN network and the network weights are adjusted until the network learns the most relevant discriminative features for the classification task.

3.4 Data Normalization

The dataset was normalized in order to make convergence faster while training the network. Data normalization is carried out by subtracting the mean from each pixel, and then dividing the result by the standard deviation [4]. Thus, each input parameter would maintain a similar data distribution. For this purpose, the standard min-max scaling was used, a normalization method for scaling data to [0, 1] as follows:

$$X_{norm} = \frac{\mathrm{X} - X_{min}}{X_{max} - X_{min}}$$

Where X_{max} and X_{min} are the maximum and minimum value of feature X respectively. For categorical data contained in the dataset, entity embedding technique was used. Generally, the categorical data which have high cardinality need entity embedding to map them to low-dimensional real vectors in such a way that similar values remain close to each other [11]. This is the case for our datasets since there are many possible values for source IP addresses, destination IP addresses, source port numbers, and destination port numbers. The number of embedding dimensions are determined according to the formula [10];

$$dimension = \left\lfloor \sqrt[4]{possibe\ values} \right\rfloor$$

Where $possibe\ values$ represents the number of possible values a categorical feature can have. A categorical feature is first mapped to an integer between 0 and n-1, where n is the number of unique values that can be taken by the feature, and then encoded as a dense vector of floating point numbers according to the dimensions as calculated above. The parameters (weights) for the dense vector representation are initialized using a random uniform distribution in the range [−0.05, 0.05]. This representation is more computationally efficient, as well as holds inherent relationship information between the categorical feature, the other features in the dataset, and the label.

3.5 Dimensionality Reduction

Several studies have demonstrated that serious redundancy among the characteristic dimensions of network data as well as high correlation exists among the data of each dimension. Further, redundancy and correlation between feature dimensions not only reduce the response time of the intrusion detection system but also affect the learning efficiency of a model training process [40]. Principal component analysis (PCA) is the most normally used method for linear dimension reduction in machine learning, particularly in data analysis and pre-processing. PCA map high-dimensional data to a low-dimensional space representation by linear projection. This study applied PCA in Scikit–Learn library [27] to analyze the variance ratio of each principal component after PCA transformation.

3.6 Labelling

Class identification of the label record was numerically processed, with 0 for Benign, and 1 for SSH-Brute force, thereby enabling one-shot processing of the label in subsequent training and testing.

3.7 Experiment Setup

Deep learning modelling relies heavily on Graphics Processing Unit (GPU) with Compute Unified Device Architecture (CUDA) core enabled. The experiments were performed a server machine running on an Intel® Core™ i7-7500U @2.90 GHz processor, NVIDIA GeForce 940MX, Ms Window 10, 8GB memory machine. The experiments leveraged the popular open source TensorFlow machine learning framework [36] running on Keras backend. The Keras library provides a convenient wrapper for deep learning models to be used as classification or regression estimators. To evaluate the performance of the deep learning classifier, the following two different test cases were considered:

i). Classifying the records as either benign or SSH-Brute force attack with all features.

ii). Classifying the records as either benign or SSH-Brute force categorizing with minimal features.

3.8 Model Design & Training

Convolutional Neural Network (CNN) was used as the deep learning algorithm for model training. CNN is a neural network architecture that uses extensive weight-sharing to reduce the degrees of freedom of models that operate on spatially-correlated features [18]. CNNs are considered a subtype of discriminative deep architecture having shown satisfactory performance in processing two-dimensional data with grid-like topology [23], such as images and videos. CNNs have powerful learning ability mainly due to the use of multiple feature extraction stages (hidden layers) that can automatically learn representations from the data [15].

The basic topology of a CNN is composed of a stack of layers (learning stages) that consists of the convolutional layer, the pooling layer, and the fully connected layers. Figure 2 shows a CNN architecture. Many CNN models have a standard structure consisting of alternating convolutional layers and pooling layers. The last layers are a small number of fully-connected layers with a softmax or sigmoid classifier.

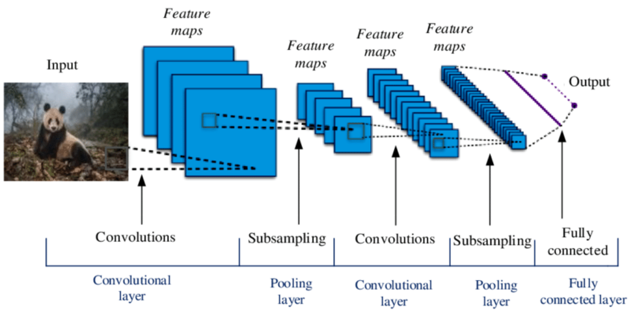

Figure 2: A Typical Convolutional Neural Network (Source:https://en.wikipedia.org/wiki/Convolutional_neural_network)

Each layer performs multiple transformations on the input data using a bank of convolutional kernels (filters). The convolution operation extracts locally correlated features by dividing the image into small slices (similar to the retina of the human eye), making it capable of learning suitable features [15]. Assuming that the input characteristic of the CNN is feature map of the layer z is $M_z(M_0 = X)$, then, the convolution process can be expressed as:

$$M_z = f(M_{z-1} \, x \, P_z + t_z)$$

Where P_z is the convolution kernel weight vector of the z layer; the operation symbol "x" represents the convolution operation; t_z is the offset vector of the z layer; and f(x) is the activation function. The convolutional layer extracts different feature information of the data matrix M_{z-1} by specifying different window values, and extracts different features M_z in the data through different convolution kernels. In addition, during the convolution operation, the same convolution kernel follows the principle of "parameter sharing" (i.e., sharing the same weight and offset) which significantly reduces the number of parameters of the entire neural network [40].

The output of the convolutional kernels is then assigned to the pooling layer which not only helps in learning abstraction but also embeds non-linearity in the feature space. The pooling layer usually samples the feature maps in accordance with different sampling rules. Assume that M_z is the input to the pooling layer and M_{z+1} is the output of the pooling layer. Then, the pooling layer can be expressed as:

$$M_{z+1} = \text{Subsampling}\,(M_z)$$

Subsampling helps in summarizing the results and also makes the input invariant to geometrical distortions. Usually, the sampling criterion selects the maximum or the mean value of the window region. In other words, the pooling layer principally reduces the dimension of the feature, thus reducing the influence of redundant features on the model. Table 2 shows the proposed model architecture that was used.

Table 2: Model architecture of the proposed convolutional neural network

Layer	Kernel Size	Stride	Output Size (Width X Length X Depth X Filters)
Input	-	-	50 X 50 X 28
Convolution Layer 1	3 X 3 X 3	1	50 X 50 X 28 X 32
Pooling Layer 1	2 X 2 X2	2	25 X 25 X14 X 32
Convolution Layer	3 X 3 X 3	1	25 X 25 X 14 X 64
Pooling Layer	2 X 2 X2	2	13 X 13 X 7 X 64
Fully Connected Layer	-	-	12,544 X 516

For the realization of the experiments, the study used the proportions 70% and 30% for the training and test datasets, respectively. Since the dataset is highly imbalanced, there was need to ensure there is similar proportion of SSH-Brute force attack records in each training and test sample as there are in the dataset as a whole. This was achieved by stratifying the dataset to ensure the distribution of benign and the malicious traffic is equivalent in both training and test data sets. In addition, a separate hold out validation set was used during the training iterations. The CNN was trained using the back-propagation mechanism [21]. The hyper-parameters used to train the model are presented in Table 3. These parameters were determined empirically according to a set of experiments carried out on the whole dataset that give the best results of classification.

Table 3: CNN training hyper-parameters

Parameter	Value
Activation	ReLu
Loss function	Cross-Entropy
Optimization algorithm	Adam
Epochs	120
Batch size	12
Learning rate	0.05
Weight decay	0.0005
Momentum	0.9
Dropout	0.2
Iterations	30

The CNN consisting of four hidden layers of 64 units per layer was designed. Channeled into these hidden layers is an initial input layer consisting of the embedded categorical variables concatenated with the statistical input features. Each layer estimates non-linear features that are passed to the next layer and the last layer in the deep learning network performed the classification. The activation function on each hidden layer was the ReLU activation function, $R(t) = max(0, t)$ and for batch normalization and regularization, a dropout rate of 0.2 was used on each of the hidden layers to obviate overfitting and speed up the model training [20]. The optimizer used is Adam, with a default learning rate of 0.05. The output layer used a sigmoid activation function and contained 1 neurons for binary classification. The loss function used was binary cross-entropy given by,

$$loss(pd, ed) = -\frac{1}{N}\sum_{i=1}^{N}[ed_i \log pd_i + (1 - ed_i)\ \log(1 - pd_i)]$$

Where pd is a vector of predicted probability for all samples in the testing dataset, and ed is a vector of expected class label, values are either 0 or 1.

3.9 Model Testing and Validation

The proposed model was used to perform a classification task. Classification is the process of assigning a label to an object based on its features translated by its descriptors. The goal is to accurately predict the target class for each case in the data. The experiment was designed to answer two questions: (i) Can a deep learning model be developed to correctly detect SSH-brute force attacks? (ii) How does the resultant deep learning model compare with other machine learning methods used in SSH-Brute force attack detection?

In order to answer the aforementioned question 1, with the features selected, a CNN model was built using the training dataset. The model was then tested and validated by performing a binary classification, classifying network traffic flows as either benign or malicious (SSH-Brute force attacks) from the test dataset. To answer the second question, the study compared the effectiveness of our deep learning model with results obtained from experiments with 5 classical machine learning algorithms, namely Naive Bayes, Logistic Regression, Decision Tree, k-Nearest Neighbour, and Support Vector Machine. Four standard metrics namely, Accuracy, Precision, Recall, and the F-measure were used for comparison.

Accuracy: It's the ratio of the correctly recognized records to the entire test dataset. In this case, the SSH-Brute force attacks. If the model accuracy is higher, the resultant model is better. Accuracy serves as a good measure for the test dataset that contains balanced classes and is defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where

True Positive (TP) - the number of SSH-Brute force records correctly classified.

True Negative (TN) - the number of Benign records correctly classified.

False Positive (FP) - the number of SSH-Brute force records wrongly classified as Benign.

False Negative (FN) - the number of Benign records wrongly classified to the SSH-Brute force records

Precision: It estimates the ratio of the correctly identified SSH-Brute force records to the number of all identified SSH-Brute force records. If the Precision is higher, the resultant model is better. Precision is defined as follows:

$$Precision = \frac{TP}{TP + FP}$$

Recall: It is also referred as True Positive Rate (TPR). It estimates the ratio of the correctly classified SSH-Brute force records to the total number of attack records. If the TPR is higher, the resultant model is better. TPR is defined as follows:

$$Recall = \frac{TP}{TP + FN}$$

F1 - Measure: F1-Measure, also called F1-Score, is the harmonic mean of Precision and Recall. If the F1-Score is higher, the resultant model is better. F1-Measure is defined as follows:

$$F1 - Measure = 2 \text{ x}\left(\frac{Precision * Recall}{Precision + Recall}\right)$$

4. RESULTS AND DISCUSSION

The proposed CNN-based model was used to classify SSH-Brute force attacks based on the images contained in the testing dataset. The fully connected layer was used where each neuron provide a full connection to all learned feature maps issued from the previous layer in the CNN. These connected layers are based on the sigmoid activation function in order to compute the classes' scores. Using TensorFlow visualization on both training and testing dataset, we can view the accuracy of our approach as shown in Figure 3.

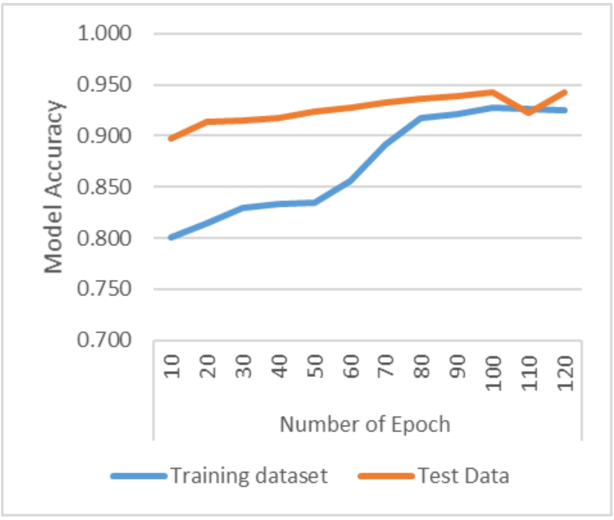

Figure 3: Model accuracy during the training and testing of the network

As shown, the accuracy of the trained data increases as number of steps (epochs) is increasing, until it reaches approximately 94 % of accuracy, which means that there is a change of 94% of detecting any SSH-Brute force attack destined towards the network. The final experimental results of the binary classification using all the features are reported in Table 4.

Table 4: Experimental results using all features

Class	Metric			
	Accuracy	Precision	Recall	F1-Measure
Benign	0.874	0.931	0.967	0.918
SSH-Brute Force	0.943	0.925	0.978	0.918

The results indicate that the model proposed in this study was able to classify SSH-Brute force attacks with 94.3% accuracy, a precision rate of 92.5%, recall rate of 97.8% and F1-score of 91.8%. Table 5 shows the classification using minimal features.

Table 5: Experimental results using minimal features

Class	Metric			
	Accuracy	Precision	Recall	F1-Measure
Benign	0.829	0.883	0.901	0.921
SSH-Brute Force	0.852	0.891	0.922	0.894

The results indicate that when minimal features were used for classification task the model performance was lower in all the metrics compared with the classification task using all the features. Using the minimal features to classify SSH-Brute force attacks, the model achieved 85.2% accuracy, a precision rate of 89.2%, recall rate of 92.2% and F1-score of 89.4%.

Finally, we compared the performance of our proposed CNN-based model with the results of the 5 classical machine learning algorithms, namely Naive Bayes, Logistic Regression, Decision Tree, k-Nearest Neighbour, and Support Vector Machine which used the same dataset.

The comparison of the classification results obtained are reported in figure 4 which shows that the deep learning based model has a better classification accuracy, recall and precision than the other machine learning algorithms. In addition, the F1 score of the model was slightly better, compared with the k-Nearest Neighbour, Logistic Regression and Support Vector Machine models.

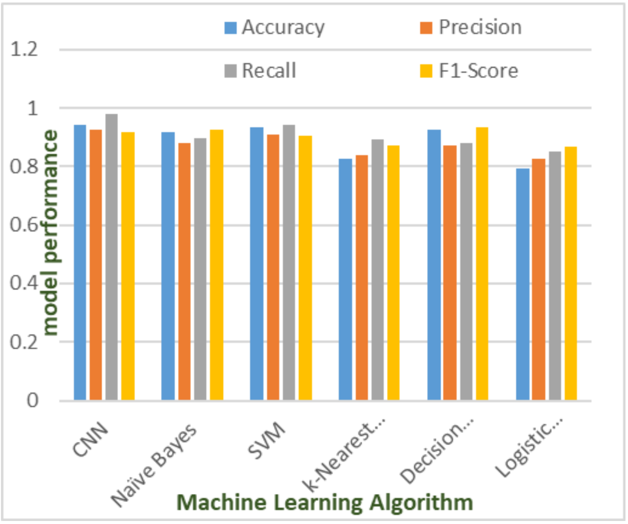

Figure 4: Performance comparison between the classical machine learning models and the CNN-based model.

The above experimental results demonstrates that the convolutional neural network model is superior to the traditional machine learning methods in terms of the ability to detect SSH-Brute force attacks. Utilizing the CICIDS dataset, and the features selected, we found success in classifying unknown network flows into benign and SSH-Brute force attacks. From just these features, we can see that the CNN-based model could identify the traits which characterize an SSH-Brute force attack.

SSH brute force attacks are common in network where an attacker attempts to guess the username and password of a user on the Secure Shell protocol. This type of network attack is simple to perform, with the results from a successfully compromised system triggering a number of destructive outcomes. Previous studies have also demonstrated that deep learning algorithms, particularly, convolutional neural networks, are effective in detecting and preventing these kinds of attacks as an alternative to the firewall techniques used today [6],[16],[22].

5. CONCLUSION AND FUTURE WORK

In this study, we proposed an approach for SSH-Brute force network attacks detection based on a supervised deep learning algorithm. A CNN-based model was designed and tested with the CIC-IDS 2018 dataset that was pre-processed for our experiment. The raw data was converted into images and then used for model training and testing. To evaluate the performance of the resultant model, two different test cases were considered; classifying the network connection record as either benign or SSH-Brute force attack with either all the features or with minimal features. The experimental results showed that our model detects benign and SSH-Brute force attack with higher accuracy and precision when all the features in are used.

Our model was further compared with experimental results obtained from 5 classical machine learning algorithms, namely Naive Bayes, Logistic Regression, Decision Tree, k-Nearest Neighbour, and Support Vector Machine. The results demonstrated that our deep learning model performed better in terms of classification accuracy, recall and precision than the other machine learning algorithms. In addition, the F1-score of the CNN based model was slightly better, compared with the k-Nearest Neighbour, Logistic Regression and Support Vector Machine models.

In the future, further experiments can be performed with other deep learning algorithms such as Deep Belief Network (DBN), Generative Adversarial Network (GAN) and the results compared with our model. In addition, our model can be tested on a different benchmark dataset such as the ISCX IDS 2012 dataset.

6. REFERENCES

[1]. AL-Zwuiany, M., & Dongjun, H. (2015). DBFST: Detecting Distributed Brute Force Attack on a Single Target. *International Journal of Scientific & Engineering Research, 6*(3), 738-744.

[2]. CICFlowMeter. (2019). *CICFLOWMETER*. Retrieved from netflowmeter.ca: https://www.unb.ca/cic/research/applications.html#CICFlowMeter

[3]. Constantin, L. (2013, November 20). *GitHub bans weak passwords after brute-force attack results in compromised accounts*. Retrieved from PCWorld.com: https://www.pcworld.com/article/2065340/github-bans-weak-passwords-after-bruteforce-attack-results-in-compromised-accounts.html

[4]. Dhakal, A., & Shakya, S. (2018, July). Image-Based Plant Disease Detection with Deep Learning . *International Journal of Computer Trends and Technology (IJCTT), 61*(1), 26-29.

[5]. Faust, J. (2018). Distributed Analysis of SSH Brute Force and Dictionary Based Attacks. *Culminating Projects in Information Assurance, 56*. Retrieved March 19, 2020, from https://repository.stcloudstate.edu/msia_etds/56

[6]. Fernandez, G., & Xu, S. (2019, October 5). A Case Study on Using Deep Learning for Network Intrusion Detection. *arXiv Preprints*, arXiv:1910.02203v1.

[7]. Gautam, T., & Jain, A. (2015, November 10-11). Analysis of Brute Force Attack using TG – Dataset. *SAI Intelligent Systems Conference 2015*.

[8]. Giménez, C., Villegas, A., & Marañón, G. (2012). *HTTP Dataset CSIC 2010*. CSIC (Spanish Research National Council). Retrieved from http://www.isi.csic.es/dataset/

[9]. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. Retrieved from http://www.deeplearningbook.org

[10]. Google. (2019). *Machine learning crash course: Embeddings*. Retrieved from developers.google.com: https://developers.google.com/machine-learning/crash-course/

[11]. Guo, C., & Berkhahn, F. (2016). Entity embeddings of categorical variables. *arXiv preprint*, arXiv:1604.06737.

[12]. Hall, M. (2000). Correlation-based feature selection for discrete and numeric class machine learning. *Proceedings of the 17th International Conference on Machine Learning (ICML '00)* (pp. 359-366). San Francisco, Calif, USA: Morgan Kaufmann.

[13]. Hira, Z., & Gillies, D. (2015). A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data. *Advances in Bioinformatics, 2015*, 1-14. doi:10.1155/2015/198363

[14]. Javedy, M., & Paxson, V. (2013). Detecting Stealthy, Distributed SSH Brute-Forcing. *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (pp. 85-96). Berlin, Germany: ACM. doi:10.1145/2508859.2516719.

[15]. Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. (2019). A Survey of the Recent Architectures of Deep Convolutional Neural Networks. *arXiv.org*, arXiv:1901.06032.

[16]. Kim, J., Shin, Y., & Choi, E. (2019, December). An Intrusion Detection Model based on a Convolutional Neural Network. *Journal of Multimedia Information System, 6*(4), 165-172.

[17]. Kumar, M. (2013, April 12). *Massive Brute-force attack Targets Wordpress sites worldwide*. Retrieved March 16, 2020, from Thehackernews.com: https://thehackernews.com/2013/04/massive-brute-force-attack-targets.html

[18]. Lai, M. (2015). Deep Learning for Medical Image Segmentation, . *arXiv.org*, p. arXiv: 1505.02000.

[19]. Lazarevic, A., Banerjee, A., Chandola, V., Kumar, V., & Srivastava, J. (2008). Data Mining for Anomaly Detection. *Tutorial at the European Conference on Principles and Practices of Knowledge Discovery in Databases*.

[20]. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436.

[21]. Leung, H., & Haykin, S. (1991, September). The complex backpropagation algorithm. *IEEE Trans. Signal Process, 39*(9), 2101-2104.

[22]. Liu, J., Song, X., Zhou, Y., Peng, X., Zhang, Y., Liu, P., & Wu, D. (2019). Deep Anomaly Detection in Packet Payload. *arXiv Preprints*, arXiv:1912.02549v1.

[23]. Liu, W., Wang, L., Liu, X., Zeng, N., & Liu, Y. (2017). A Survey of Deep Neural Network Architectures and Their Applications. *Neurocomputing, 234*, 11-26. doi:10.1016/j.neucom.2016.12.038

[24]. LookingGlassCyber. (2017, October 5). *Protecting Your Network Against Brute Force Password Attacks*. Retrieved from lookingglasscyber.com: https://www.lookingglasscyber.com/blog/threat-intelligence-insights/protecting-network-brute-force-password-attacks/

[25]. Mohammed, M., Degadzor, A., Effrim, B., & Appiah, K. (2017). BRUTE FORCE ATTACK DETECTION AND PREVENTION ON A NETWORK USING WIRESHARK ANALYSIS. *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY(IJESRT), 6*(6), 26-37. doi:10.5281/zenodo.802797

[26]. Najafabadi, M., Khoshgoftaar, T., Calvert, C., & Kemp, C. (2015). Detection of SSH Brute Force Attacks Using Aggregated Netflow Data. *IEEE 14th International Conference on Machine Learning and Applications*, 283-288. doi:10.1109/ICMLA.2015.20

[27]. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., & Thirion, B. (2011, October). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12*, 2825-2830.

[28]. Saito, S., Maruhashi, K., Takenaka, M., & Torri, S. (2016, March). TOPASE: Detection and Prevention of

Brute Force Attacks with Disciplined IPs from IDS Logs. *Journal of Information Processing, 24*(2), 217-226.

[29]. Sangkatsanee, P., Wattanapongsakorn, N., & Charnsripinyo, C. (2011). Practical real-time intrusion detection using machine learning approaches," , vol. 34, no. 18, pp. *Computer Communications*, 2227–2235. doi:10.1016/j.comcom.2011.07.001

[30]. Schwartz, M. (2017, July 24). *Mirai Malware Hacker Pleads Guilty in German Court*. Retrieved from bankinfosecurity.com: https://www.bankinfosecurity.com/mirai-malware-hacker-pleads-guilty-in-german-court-a-10140

[31]. Seals, T. (2016, February 8). Massive Brute-Force Attack on Alibaba Affects Millions. *Infosecurity Magazine*. Retrieved March 16, 2020, from https://www.infosecurity-magazine.com/news/massive-bruteforce-attack-on/

[32]. Seals, T. (2017, July 25). Widespread, Brute-Force, Cloud-to-Cloud Attacks Hit Office 365 Users. *Infosecurity Magazine*. Retrieved March 16, 2020, from https://www.infosecurity-magazine.com/news/widespread-bruteforce-office-365/

[33]. Sharafaldin, I., Lashkari, H., & Ghorbani, A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. *4th International Conference on Information Systems Security and Privacy*, (pp. 108-116).

[34]. Shiravi, A., Shiravi, H., Tavallaee, M., & Ghorbani, A. (2012, May). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computer Security, 31*, 357-374.

[35]. Taher, K., Jisan, B., & Rahman, M. (2019). Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection. *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)* (pp. 643-646). IEEE.

[36]. TensorFlow. (2019). *TensorFlow: An end-to-end open source machine learning platform.* Retrieved from TensorFlow.org: https://www.tensorflow.org/about/

[37]. Tsai, C., Hsu, Y., Lin, C., & Lin, W. (2009). Intrusion detection by machine learning: A review. *Expert Systems with Applications, 36*(10), 11 994–12 000. doi:10.1016/j.eswa.2009.05.029

[38]. Verizon. (2018). *2018 Data Breach Investigations Report.* Verizon.

[39]. Vinayakumar, R., Alazab, M., Soman, K., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access, 7*, 41525-41550. doi:10.1109/ACCESS.2019.2895334

[40]. Xiao, Y., Xing, C., Zhang, T., & Zhao, Z. (2019). An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks. *IEEE Access, 7*, 42210-42219.

[41]. Yuvaraj, M., Bharathidasan, A., & Kumar, N. (2014). Implementation of Password Guessing Resistant Protocol (PGRP) to Prevent Online Attacks. *International Journal of Computer Science and Mobile Computing (IJCSMC), 3*(2), 815-826.

Stephen Kahara Wanjau received his B.Sc. Degree in Information Sciences from Moi University, Kenya, in 2006 and MSc. Degree in Computer Systems from Jomo Kenyatta University of Agriculture and Technology, Kenya, in 2018. Currently, he is pursuing a PhD in Computer Science at Murang'a University of Technology. He is currently serving as the Director of ICT at Murang'a University of Technology, Kenya. His research interests include machine learning, network security, big data analytics, knowledge management, and cloud computing.

Geoffrey Mariga Wambugu received his B.Sc. degree in Mathematics and Computer Science from Jomo Kenyatta University of Agriculture and Technology (JKUAT), Juja, Kenya, in 2000, the M.Sc. degree in Information Systems from The University of Nairobi, Nairobi, Kenya, in 2012, and the Ph.D. degree in Information Technology JKUAT, in 2019.
He have served for over 10 years' as head of department in higher education institutions in Kenya and also been involved in the design, development, review and implementation of Computing Curricula in different universities in Kenya. Currently he is a Lecturer and Information Technology head in Murang'a University of Technology. His research interests include Probabilistic Machine Learning, Text Analytics, Natural Language Processing, Data mining, Big Data Analytics. At present, He is engaged in university teaching and research supervision.

Gabriel Ndung'u Kamau received his Bed (Art) degree in Mathematics and Business in 1999 from Kenyatta University and Master of Business Administration (Management Information Systems) in 2008 and PhD in Strategic Information System in 2017 from University of Nairobi. Gabriel is also a Certified Network Security Specialist (2020) and Big Data Analyst (2019). Gabriel was a teaching assistant lecturer with Department of Computer and Information Technology, Kenya Methodist University from 2009 to April 2013. In June, 2013, Gabriel Joined Murang'a University of Technology as a lecturer in the department of Information Technology. Gabriel has a vast knowledge and teaching experience in the area of management information systems, information security and applied cryptography, computer forensics, enterprise risk management of information systems, and IT Governance.
His research interest includes ICT4D, cybersecurity and forensics, data analytics, computing and information technology philosophy perspectives.