International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 8, ISSN No. 2455-2143, Pages 148-159
Published Online December 2019 in IJEAST (http://www.ijeast.com)

DEEP LEARNING APPLICATIONS IN CYBER SECURITY: A COMPREHENSIVE REVIEW, CHALLENGES AND PROSPECTS

Bhavuk Sharma
Department of Computer Engineering
DJSCE Mumbai, Maharashtra, India

Dr Ramchandra Mangrulkar
Department of Computer Engineering
DJSCE Mumbai, Maharashtra, India

***Abstract*— Deep Learning applications are starting to pervade every commercial and technological sector. This paper surveys deep learning (DL) methods for cyber security applications, highlights the security considerations when using deep learning networks and presents possible malicious uses of such models. Initially, a brief description of the architectures studied and presented in this paper is provided. In the next section, the security applications are indicated for different models including CNNs, RNNs and GANs. Lastly, security is-sues are divided into two sections: attacks on various deep learning net-works, and attacks on a network using Deep Learning models themselves.**

Keywords— **Cyber Security, Deep Learning, Neural Networks, Cyber attacks**

I. INTRODUCTION

Since its arrival, Deep Learning has gained popularity for use in artificial intelligence applications due to the variety of neural networks in different areas of interest. The usefulness of Deep Learning comes from its ability to work on both labelled and unlabeled data as well as different types of data such as numbers, text or images. Particularly, in the field of cybersecurity, there is a significant use for AI solutions, e.g. anomaly detection, malware identification, intrusion detection systems etc. These applications help overcome the need for human labor and also act as analyst agents when human analysis is not sufficient.

Deep learning (DL) is a subset of machine learning algorithms that are based on artificial neural networks. These networks consist of multiple layers, including input and output layers, and each layer has multiple units known as neurons, that perform weighted functions. DL consists of various architectures for different applications, such as feedforward, convolutional and recurrent networks. The usefulness of deep learning was recognized recently in the years 2011-2012, when a fast GPU implementation of CNNs [1] was achieved and it set bench-marks in computer vision contests. CNNs gained widespread recognition after the victory in the "ImageNet Large Scale Visual Recognition Challenge" by the CNN architecture Alexnet [2], which won by a significant margin compared to other models. Deep Learning applications have also been applied to sequential data analysis, pattern recognition and natural language processing (NLP) tasks that are relevant for cybersecurity operations.

Cybersecurity (or information security) is defined by the ITU (International Telecommunication Union) as follows: "Cybersecurity is the collection of tools, policies, security concepts, security safeguards, guidelines, risk management approaches, actions, training, best practices, assurance and technologies that can be used to protect the cyber environment and organization and user's assets. Organization and user's assets include connected computing devices, personnel, infrastructure, applications, services, telecommunications systems, and the totality of transmitted and/or stored information in the cyber environment."

With the increasing number and variety of computing devices around the world, the need to protect them as well as the ways to exploit them have also in-creased. Along with the increasing technology, the uses of Artificial Intelligence have also seen a surge. AI has an important role to play when in Cyber Security; several security applications have been built around AI, many real-world applications use AI which raises security concerns and lastly there are several ways that AI can be used for unethical cyber activities.

This paper explores the role of deep learning in cybersecurity, presenting use-fulness as well as the malicious ways it can be used to cause harm to users and organizations, along with the security issues of deep learning applications. Sub-sequent sections are described as follows: the second section provides a review of the works related to this paper; the third section outlines beneficial applications of different DL architectures; the next section delves into the misuses of AI and shows its dual nature of being beneficial as well as potentially destructive.

The paper aims to bring into light the serious issues that may arise in the near-future due to automated cybercrimes while also highlighting some helpful methods that the way to security procedures can also be automated. The paper focuses solely on Deep Learning applications and their uses and misuses in the cybersecurity sector and the security issues of real-world applications.

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 8, ISSN No. 2455-2143, Pages 148-159
Published Online December 2019 in IJEAST (http://www.ijeast.com)

II. LITERATURE REVIEW

There have been several papers that highlight the use of artificial intelligence and deep learning techniques as security measures as well as malicious cyber-weapons. [3] presents the different applications of AI, not limited to deep learning, in Cyber-security, such as Neural Networks and Intelligent Agents. It pro-vides use cases for AI in security applications.
[4] exclusively looks at deep learning methods that boost security measures against cyber-attacks. After providing a terminology of different cyber-threats and DL architectures, they surveyed different models and presented them in the categories of different security applications, such as malware detection, spam classification and ransomware.
[5] describes the malicious applications of AI and ways to predict, prevent and mitigate to them. It considers the issues and attacks on three different types of security: physical, digital and social. The report discusses each security type in depth, suggests threats in that domain and provides control points and counter-measures that should be considered during security analysis. It also gives recommendations to be followed when working with AI and a strategic analysis of the influence of AI in the future, in relation to security.
[6] provides an analysis of different machine learning techniques, not limited to deep learning, that can be used for cyber security applications. The researchers also highlighted the vulnerability of the machine learning algorithm to adversarial attacks and the need for precise parameter tuning and re-training for better results.
[7] cover the various attacks on machine learning algorithms and suggest defenses against them. The paper covers two categories of attacks, evasion and poisoning, followed by ways to defend against them. These categories are further divided into different types, which are described in detail. They also cover data privacy under AI applications.
[8] covers the adversarial attacks on the systems deploying deep learning models for the analysis of images. They review the designs, analyze the existence, and propose ways to defend against such attacks. The paper covers adversarial attacks only on networks dealing with Computer Vision such as convolutional neural networks and generative networks (variational autoencoders and GANs).

III. DEEP LEARNING ARCHITECTURES

The presented deep learning architectures are derivations of neural networks, a class of machine learning algorithms that are modelled from the neural connections in the brain. These networks consist of input and output layers and at least one hidden layer between them. Layers consists of computation units known as neurons, which take input and perform mathematical operations known as activation functions and pass the output to the next layer or represent it as the final output (in the output layer).

A. Deep Neural Networks

A Deep Neural Network is a feed forward neural network wherein connections between the nodes do not form a cycle [9], and one which contains more than one hidden layer. These networks follow a linear flow of data from the input layer to the output layer, through one or more hidden layers.
These networks are able to map complex functions given the required number of layers. DNNs are commonly used for classification tasks in various sectors such as text classification, anomaly detection and financial sales prediction.

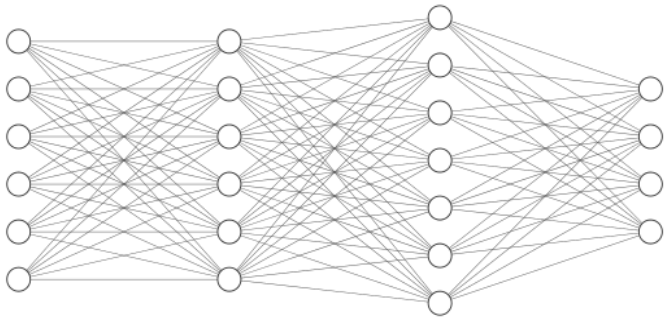

Fig. 1. Deep Neural Network

B. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep learning systems that process data in groups using functions known as kernels. Using these kernels, they are able to extract features from the data over a region, as opposed to neural networks where only single data points are analyzed. CNNs are most commonly used for Computer Vision tasks such as object classification using ImageNet Dataset [10], that has resulted in architectures such as Microsoft's ResNet[11] and Google's InceptionV3[12]
.

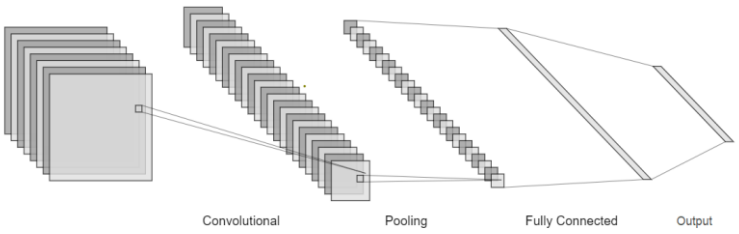

Fig. 2. Convolutional Neural Network

A typical CNN architecture, as shown in Fig 2, consists of three types of layers: convolutional, pooling, and the fully connected layer. The kernel function is performed in the convolutional layer. The kernel weights are learnt by the model during training phase, such that each kernel represents

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 8, ISSN No. 2455-2143, Pages 148-159
Published Online December 2019 in IJEAST (http://www.ijeast.com)

certain features of the data. Pooling layers are added in between convolution layers to compute an average and reduce size of the input. The fully connected layer, flattens out the output of the convolutional layer and performs classification on the features provided. It generally consists of a DNN but may also use RNNs (recurrent neural networks) to obtain hybrid deep learning architectures.

C. Recurrent Neural Networks

Recurrent Neural Networks or RNNs are deep learning models that, unlike standard DNNs, include the output of a previous input as part of the next input. Consequently, these networks are able to analyze sequential data and are used in tasks that involve sequential analysis, such as speech recognition, handwriting recognition and semantic analysis.

RNNs suffer from the vanishing gradient problem, wherein, during the backpropagation/learning phase, the collective product of small partial derivatives results in an exceedingly small value of the gradient, that causes the change in weights to halt or become negligible. To resolve this, LSTM (Long short-term memory) networks are used. These networks have a component known as a "forget gate", that decides the importance of previous information collectively, replacing the product in case of RNNs with a single value.

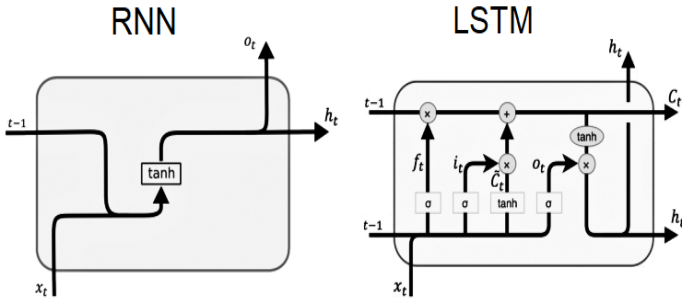

Fig. 3. RNN unit vs LSTM unit

D. Generative Adversarial Networks

GANs are generative neural networks developed by Goodfellow et al (2014) [13]. They comprise of two models trained simultaneously, a generator and a discriminator. The generator attempts to generate the desired output based on the given input, and the discriminator analyzes this output and determines how closely it resembles actual output's characteristics. Both models are trained using log loss functions and backpropagation.

GANs have become very popular since their inception, with the creation of several variants including CycleGAN [14], contextual GAN [15] and VAE-GAN [16]. Their applications are most apparent in Computer Vision and include image generation, image style transfer and 3D object generation.

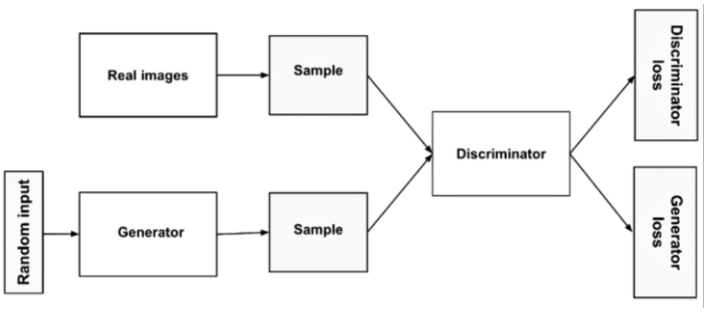

Fig. 4. GAN architecture

E. Autoencoders

An autoencoder is a neural network that aims to recreate the input at the output layer, with multiple hidden layers in between. The aim of an autoencoder is to learn a representation (encoding) for a set of data. It does so by isolating a hidden layer in the network and using it for representation of data. When the size of this layer is less than the input, the autoencoder effectively reduces the dimension of the data.

Autoencoders are used for creating encodings of data and for data compression. They can also be used for removing noise from input, and when used so are known as denoising autoencoders. Applications of autoencoders also include feature extraction, image classification and image generation [16].

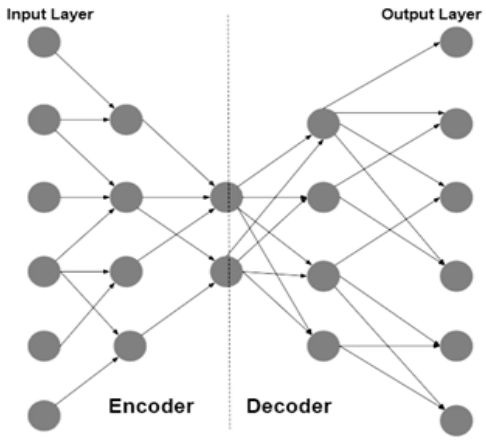

Fig. 5. Autoencoder architecture

IV. DL TECHNIQUES FOR PROTECTION

A. Deep (Feedforward) Neural Networks

The feedforward neural network is the simplest type of artificial neural network. Despite being the simplest of neural network architectures, DNNs are quite useful for performing classification and regression tasks due to their property to estimate complex functions, given enough layers. [17] In the

domain of cybersecurity, DNNs have proven to perform well in the detection of malwares, threats and misuses of technology.

Cannady proposed the using neural networks for misuse detection in [18], due to their adaptive nature to learn new patterns and ability to output probabilities instead of a definitive response. He trained a model neural network which gave a high accuracy of above 90 per cent for the selective test classes, indicating their potential for real applications.

Wu (2009) [19] used a hybrid method rule-based and neural network method for e-mail spam classification. Their model analyzed the more consistent factor in spam e-mails, intrinsic spamming behaviors, instead of keywords present in the content. The model outperformed several other contemporary networks such as NaiveBayes, ADTrees and Bayesian Networks, achieving an accuracy of 99.60%.

Salvador et al. (2009) [20] suggested a novel approach for detecting Zombie PCs using neural networks. Their approach was to analyze the network traffic using a neural network that is trained on synthetic data. The model achieved good results for various traffic scenarios, with a minimum accuracy of 87.34%.

Tuor et al. (2017) [21] used a novel architecture comprising of DNNs and RNNs, for the task of insider threat detection. Experiments showed that their models were able to outperform standard anomaly detection techniques including Principal Component Analysis and Support Vector Machines. The DNN was able to place all malicious events above 50th percentile anomaly and most above 95th percentile.

DNNs have also been deployed in intrusion detection systems (IDS). Bitter et al (2010) [22] studied the usage of DNNs in IDS. The performance of DNNs was compared with state-of-the-art for IDS, DoS attacks and Spam detection. DNNs used for IDS were able to achieve average accuracies of 83.14%. They concluded that these networks performed well, and were viable for building robust IDS in non-idealistic environments.

[23] used DNNs to identify Spectre attacks by detecting cache side-channel at-tacks. Side-channel attacks observe the changes made in the environment of a computer system to extrapolate its inner workings and exploit them. The authors analyzed the characteristics of the Hardware Performance Counters (HPCs) to observe the CPUs cache activity using neural networks, and classify cache accesses as malicious or benign. The authors were able to achieve an accuracy of 99.23%.

Finally, Vinayakumar et al. (2018) [24] applied DNNs to multiple security ap-plications. They used three different datasets, for the three applications, APKs from Opera Mobile store for malware classification, UniteCloud UTM for incident detection and unified, anonymized data using HCRUD for fraud detection. The DNN models were compared to and outperformed XGBoost in all use cases, with a minimum classification accuracy of 94%.

B. **CNNs**

Convolutional networks are generally used for image processing tasks due to their distinct kernel layers that perform the convolutions. In cybersecurity, how-ever, these systems can be used for classification tasks by converting input features into images and feeding them into the convolutional model.

Yu et al. (2017) [25] proposed a novel technique to detect domain names generated by domain generation algorithms (DGAs). DGAs are used by botnets to create communication links for their command and control servers. The authors trained two types of deep networks: CNN and an LSTM, each with an embedding layer, and analysed real massive traffic for real-time DGA detection. AUC achieved by CNN and LSTM were 0.9918 and 0.9896. The authors noted that the CNN performed best out of all models at 0.01% FPR.

Zeng et al. (2017) [26] performed DGA detection using multiple models pre-trained on ImageNet including AlexNet[2], Inception[12] and ResNet[11]. The models were adjusted to work with string inputs, re-trained and then its output was fed into a decision tree classifier. They achieved the best results with an ac-curacy of 99.56%, 99.86% TPR and a 1.128% FPR using the Inception V4 model.

Hendler et al (2018) [27] addressed the issue of Powershell (Microsoft's command line shell) attacks by detecting malicious Powershell commands using NLP, CNN, and LSTM based classifiers along with an ensemble. By creating an ensemble, they were able to achieve the best results over the non-ensemble methods, obtaining TPRs 0.92, 0.89 and 0.72 for FPRs of 10-2, 10-3 and 10-4.

Shibahara et al. (2017) [28] proposed an EDCNN architecture to detect drive-by download attacks. These attacks inject malicious code into the victim's user when they visit an affected website or download affected software. The event denoising convolutional neural network (EDCNN) takes URL sequences as inputs and outputs whether it is malicious. The EDCNN was able to classify the sequences with a low FPR, detect malicious redirections and take the least computation time.

Wang et al. (2017) [29] built an intrusion detection algorithm using raw net-work traffic data which were then fed into a CNN. The classification was achieved in two ways: firstly, using a 20-class classifier and 10-class classifier, that output either malicious or a type of non-malicious class. Secondly, they used binary classification to output either malicious or benign. They achieved an ac-curacy of 99.17% with the 20-class classification and 99.41% with 10-class classification while the binary classifier achieved 100% accuracy for the involved datasets.

C. **RNNs**

RNNs, esp. LSTMs, have become widely popular for sequential analysis of data. Cybersecurity applications for LSTMs include intrusion detection and anomaly detection

since they are able to detect continuous patterns such as anomalous behaviour or malicious attacks.

Kolosnjaji et al. (2016) [30] used CNN and RNN hybrid to identify malware. They converted the call sequences into one-hot encoding which were then used to train the DL model. The model consisted of a CNN and an LSTM with the out-put function softmax. It achieved an accuracy of 89.4%, and its accuracy, precision and recall were better than those Support Vector Machines and Hidden Markov Models.

Torres et al. (2016) [31] proposed an LSTM with 128 nodes for botnet detection. The features were one-hot encoded and the model was trained with stratified 10-fold cross validation. The data was selected in three ways: oversampling, un-der sampling and without sampling. The oversampling method provided the best results: 96% accuracy and an average false alarm rate of 0.0111. McDermott et al. (2018) [32] trained a bidirectional LSTM to detect botnets in IoT devices. The LSTM model consisted of a word-embedding layer to create a representation of the string data input to the model. The authors built their own dataset of Mirai botnet traffic, and tested the model on four types of attacks used by Mirai: User Datagram Protocol (UDP) flood, Acknowledgment (ACK) flood, Domain Name System (DNS), and Synchronize (SYN). The bidirectional LSTM achieved accuracies ranging from 91.95% accuracy to a maximum of 99.999% accuracy based on the type of attack.

There have been several works that have tested the use of RNNs for intrusion detection systems. Staudemeyer (2015) [33] used LSTMs and Kim and Kim (2015) [34] used RNNs on the KDD-1999 datasets, and Kim et al. (2016) [35] used an ensemble of LSTM classifiers on the KDD-1999 and additional data they generated. The best results were obtained by Kim and Kim [34], achieving a 100% detection rate with a 2.3% false alarm rate.

Loukas et al. (2017) [36] used LSTMs to detect various attack types including DDoS, command injection, and network malware. They achieved an accuracy of 86.9%, which was better than other algorithms that were tested. The LSTM also outperformed other models when tested with unknown data.

Maniath et al (2017) [37] proposed a method to classify API calls as ransom-ware. They performed dynamic analysis on executables' API calls, which were integer encoded, assigned to a ransomware or benign class and then fed into the LSTM network. With a dataset of 157 different ransomware samples of variable length, they were able to achieve a high accuracy of 96.67%.

There have been several works that propose the use of RNNs for authentication in IoT (Internet of Things) applications. Ferdowsi et al (2018) [38] proposed a novel algorithm that watermarked the signals of an IoT device using an LSTM model that uses stochastics features such as the signal's skewness or spectral flatness. They used this watermarked signal for secure signal authentication and furthermore detect any attack in the system. Kong et al (2019) [39] propose a real-time authentication and protection mechanism for Smart homes, which they named Fingerpass. It authenticates the user by deriving the unique behavioural characteristics of finger gestures from channel state information of WiFi signals. The LSTM network was used for user authentication during the login phase, and achieved an average accuracy of 92.6% and average FPR of 3.8%, along with a satisfactory latency of 800-1200ms.

D. **GANs**

GANs are used to generate images, patterns or any other data using their genera-tor-discriminator architecture. This ability of data generation has been observed to lead to more malicious use-cases than beneficial ones, which will be discussed in the next section. However, there have been certain positive applications of GANs for security applications, as stated below:

Zenati et al (2018) [40] developed a GAN model for anomaly detection. They had modified the GAN model such that it created a latent representation using an encoder, apart from using a discriminator and a generator. During the training phase, this latent representation was learned as well and was used by the discriminator alongside generator output. It was then evaluated on both visual (MNIST) and non-visual (KDD-99) data. Their model performed at par with state-of-the-art, achieving 92% precision, 95.82% recall and an F1 score of 93.72%.

Chen et al (2019) [41] used a GAN based model with multiple intermediate layers for intrusion detection. They modelled their architecture on the bidirectional GAN (Bi-GAN) and tested it on the KDD-99 dataset. Their model outperformed state-of-the-art models with 93.24% precision, 94.73% recall and an F1 score of 93.98%. It also reduced the overhead by reducing training time by 12% from the state-of-the-art and speeding up testing by a factor of 1.357, compared to Bi-GAN.

Volkhonskiy et al (2017) [42] proposed the use of GANs for the purpose of steganography in images. They used Deep Convolutional Generative Adversarial Networks (DCGANs) for generating the covers of images. They showed that the DCGAN is able to effectively create cover images and that their network can also be used for image encryption, with a reconstruction quality of almost 1, at a minimum of 98.65%.

Shi et al (2017) [43] proposed a novel strategy of using GANs for performing secure steganography. Their proposed model, SSGAN, has one generator and two discriminators. The generator G is used to create covers for steganography, discriminator D is used to assess image quality and discriminator S is used for performing steganalysis on the image to judge its suitability. The models were trained using the CelebA face dataset [59]. They showed an im-proved convergence time over their counterpart, SGAN [42]. Their model was found to increase error rate during steganalysis, which showed that their model had created a more secure cover.

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 8, ISSN No. 2455-2143, Pages 148-159
Published Online December 2019 in IJEAST (http://www.ijeast.com)

Lee et al. (2017) [44] used GANs to train a network that is robust to adversarial inputs. They train a generator to create adversarial examples and simultaneously train a classifier that is able to correctly classify these adversarial examples, in addition to the original images. The authors used the CIFAR 10 and CIFAR 100 datasets, and their classifier showed the best results out of all methods (dropout, adversarial training and random perturbation). The authors noted that their model provided better generalization than the fast gradient method, but it was also slower and needed further study for improved performance.

Hayes et al (2017) [45] present an issue with generating images using generative models, information leakage, and a method to avoid the same. Information leak-age can allow the attacker to deduce information about the dataset used to train them. The authors propose the use of a GAN model to detect overfitting in a generative network, following which they compare the differences between the real and synthetic images generated, that can be used to determine the data it was trained on. For the CIFAR-10 dataset, they were able to achieve accuracies of 100% for white-box, 58% with black-box and limited knowledge and 37% with black-box attacks and no knowledge. These tests show that the creators of these generative networks can use the authors' model for detecting overfitting in their networks.

Yin et al (2018) [46] propose Bot-GAN, an architecture that is used to improve the performance of botnet detection models by generating fake samples. The Bot-GAN was trained on the ICSX botnet dataset, which contained four types of bot-net traffic. It generates samples of 122 features which belong to one of the four classes mentioned. Experiment results showed that the model was able to improve the performance of the original detector. When using 500 mixed samples, their FPR dropped from 19.19% to 15.59%.

E. **Autoencoders**

Autoencoders have shown to be able to perform well in intrusion detection systems and classification tasks. The ability reduce data to lower dimensions allows them to be used in hybrid models in conjunction with other DL architectures such as RNNs and CNNs.

Wang and Yiu (2017) [47] performed malware classification using RNN-based autoencoders. They used the autoencoders to reduce the dimensionality of API call sequences, and then passed them onto a RNN for malware classification. They achieved 99.1% accuracy using a public malware dataset. They also used the model for representation learning of file access patterns (FAP) and achieved 98.3% accuracy for FAP generation.

Lotfollahi et al. (2017) [48] used CNN and stacked autoencoder (SAE) in a novel architecture called Deep Packet, to classify traffic and identify applications. The authors achieved an F1 scores of 95% and 92% for application identification and traffic characterization respectively. The authors concluded that their model can be used to automate feature extraction and classification of traffic and upon further study, used for classifying Tor traffic.

Aminanto and Kim (2017) [49] trained a stacked autoencoder to detect impersonation attacks. These attacks involve a device that pretends to be a legitimate access point and gains access to the network to perform malicious activities. They used the "CLS" dataset of the AegeanWiFi Intrusion Dataset (AWID) [50], in which they considered two classes out of four, benign and attack only. The stacked autoencoder consisted of two hidden layers, and was used to extract features from the dataset. These features were then passed on to a k-means clustering algorithm to separate into impersonating and real classes. They achieved an accuracy of 94.81%, with a recall of 92.18% and an 86.15% precision.

Sun et al (2018) [51] proposed a method for anomaly detection using variational autoencoders (VAE) on various datasets. They use the KDD-CUP'99 dataset for network anomaly detection, the MNIST dataset for image anomaly screening and the UCSD Pedestrians dataset for anomaly detection in video surveillance. To detect anomalies in the input, they trained the network on the dataset and learned to map the hidden features onto a latent variable, which was then used to reconstruct output for test data. The degree of difference between the reconstructed output and test input represented the abnormality and was used to classify data as anomalous. For the KDD dataset, their model performed better than or at par with the best detectors, achieving an AUC of 95.1% and an F1 score of 81.2%.

Ma et al. (2018) [52] performed intrusion detection on sensor networks, using a novel approach called SCDNN. The SCDNN classifier used a combination of spectral clustering and deep neural networks to classify intrusion data features into five classes: Normal and four malicious classes. SCDNN uses spectral clustering (SC) to create k subsets/ clusters which are trained using k deep neural networks (DNNs). The DNN architecture consists of autoencoder stacked together in the hidden layers. The authors tested the model for different cluster sizes k, and noticed the best results (most stable) with k=2.

Diro and Chilamkurti [53] performed intrusion detection on IoT networks and devices, using a 3-layer autoencoder. They aimed to detect DoS (denial of ser-vice) and other types of attacks on fog-to-things/edge computing systems by as-signing parameters to each worker fog node as input and updating the former via a master node. The model used for training was a stacked autoencoder architecture, trained in an unsupervised manner. Using the KDDCUP99 dataset, they were able achieve a 99.2% accuracy, compared to a 95.2% accuracy from shallow learning algorithms.

Alam et al (2019) [54], developed a ransomware detection system, RATAFIA, using LSTM-based autoencoders trained on normal program behaviours. They analysed the hardware changes made by an executable, obtained from the Hard-ware Performance Counter (HPC) statistics. The detection is performed in time windows that are decided by the

autoencoders and differ for each processor type. They found that their system was able to detect the WannaCry ransomware in 5.313 seconds, and whereas other tested ransomwares were detected in less than 2 seconds. Also, RATAFIA is able to work with any modern processor that sup-ports HPC, without any kernel modifications.

Similar to [39], Shi et al. (2017) [55] used WiFi signals generated by IoT de-vices to detect human behavioural and physiological features based on their daily activity patterns using an autoencoder. The autoencoder uses this channel state information (CSI) to create a unique identification ("fingerprint") for each user. The autoencoder consisted of three layers for different functions: first layer for separating out different activities, second layer for action classification and third layer for identification of users based on their fingerprint. The architecture also consisted of an SVM layer to detect spoofing. The model achieved an authentication accuracy of 91%.

V. SECURITY ISSUES

Along with useful applications in security, Deep Learning models also carry potential threats with them, which must be taken into account when implementing them into a security system. Also, Deep Learning models also have certain properties that allow them to be used for harmful applications, as stated in section B.

A. Attacks on Deep Neural Networks

Attacks on Deep Learning architectures can be divided into two categories: adversarial and poisoning attacks. Adversarial attacks are used to deceive machine learning algorithms into misclassifying input. It is achieved by adding noise to the input that is not perceivable to humans, but greatly changes the model's out-put. Poisoning attacks aim at the training phase of the model, wherein they affect the model's learning ability by changing the data so that the model performs bad-ly or classifies according to the attacker's choice.

Tabacof et al. (2016) [56] explored adversarial attacks on variational autoencoders, and proposed a technique to create adversarial examples, by distorting input images, that were able to fool the autoencoder into misclassification. Their methodology aimed to distort the input such that its representation by the auto-encoder was similar to the target image. The study concluded that there was a linear trade-off for such attacks: adversarial examples for autoencoders require more than small distortions to make a significant impact on output.

Kos et al. (2018) [57] also explored adversarial attacks on the variational autoencoder (VAE) and the VAE-Generative Adversarial Networks (VAE-GANs). The authors presented three different types of attacks on VAEs and VAEGANs: using a classifier for manipulation, exploiting the loss function of VAE and targeting latent layers of the generative models. The authors tested the attacks on MNIST and SVHN datasets and found that the Latent attack gave the best results but was also the slowest, whereas the Classifier performed the worst but was the fastest. The Loss attack had a comparable time-to-attack with the Classifier at-tack. The authors stated that their work proved that adversarial examples were a general phenomenon in generative models and provided a foundation for under-standing and building more robust models.

Rozsa et al. (2019) [58] introduced a novel FFA (fast flipping attribute) technique to generate adversarial examples on the CelebA dataset [59]. FFA worked by zeroing out input attributes that result in the target image at the output, and computing and adding perturbations in the input accordingly. The FFA algorithm proved to create more robust example compared to the fast gradient sign (FGS) method. The paper also discussed naturally occurring adversarial examples and proposed a novel way to address their misclassification. The authors concluded that to make the DNN more robust it should be trained with adversarial examples and extreme positives.

Papernot et al. (2017) [60] demonstrated one of the first attacks against the deep neural network classifiers in cyberspace in the real-world settings. They performed a black-box attack on the targets: remotely hosted neural networks by MetaMind (an online deep learning API), Amazon and Google. They showed that these classifiers had error rates of 84:24%, 96:19% and 88:94% respectively, for the generated adversarial examples. The authors also showed that their black-box attack was able to evade gradient masking technique, increasing the resilience of adversarial attacks.

Ilyas et al. (2018) [61] also developed adversarial attacks aimed at real-world classifiers that were more restrictive than typical black-box models. They developed threat models for three real-world settings: query-limited, partial-information and label-limited. The attacks were proved to be effective under these settings, with successful prediction of the target class in 99.2%, 93.6% and 90% test cases respectively. Their attacks were also able to deceive the Google Cloud Vision API using the partial-information attack.

Athalye et al. (2017) [62] generated robust adversarial examples for 2D as well as 3D object classifiers. They demonstrated the existence of robust 3D adversarial objects, and introduced the first algorithm for synthesizing examples that are adversarial over a chosen distribution of transformations. They use the Expectation Over Transformation (EOT) algorithm to create adversarial examples that remain adversarial after a chosen transformation. To develop a 3D adversarial object, they used transformation functions that rendered 3d models for given textures and applied EOT to generate adversarial textures for the real world. They were able to achieve high "adversariality" (percentage of samples classified as target class): 96.4% in 2D and 83.4% in 3D classifiers.

Yang et al (2017) [63] test the applicability of the traditional direct gradient method to generate poisoned data against neural networks. They also propose a generative method which uses an autoencoder to generate the poisoned data. The autoencoder gets updated by a reward function of the loss, and

the target model receives this poisoned data to calculate the loss with respect to. the normal data. The authors tested this on MNIST and CIFAR datasets. Their results show that the generative method can speed up the poisoned data generation rate almost by a factor of 240, compared with the direct gradient method, at the cost of a slightly lower model accuracy in the MNIST dataset. A countermeasure is also designed to detect such poisoning attack methods by checking the loss of the target model. They also provide a countermeasure to the attack, by checking the value of the loss periodically, we can detect if the data is poisoned if loss exceeds a certain threshold.

Muñoz-González et al (2017) [64] explore poisoning attack methods for multiclass problems. They propose a novel approach towards data poisoning, which exploits back-gradient optimization, wherein they trace the backpropagation during gradient ascent and compute the poisoning gradient in an incomplete inner optimization cycle. The algorithm has a linear complexity dependent on number of iterations, which makes it computationally efficient. The authors evaluate its effectiveness on multiple applications including spam filtering, malware detection, and handwritten digit recognition. They noted that neural networks could be compromised by even a small number of adversarial points.

Chen et al (2017) [65] investigated backdoor poisoning strategies on deep networks. A backdoor is essentially a specific trigger for a software technology that allows the user to bypass restrictions or access hidden/restricted features. In the context of deep learning, attackers can create backdoor keys (unique image attributes) by poisoning the model during training, which can be used in the fu-ture to make the network misclassify the data as the attacker's target class. Chen et al studied poisoning strategies which could work even under strict constraints such as: the adversary has no knowledge of the model and the training set used by the victim system; or, the attacker is allowed to inject only a small number of poisoning samples. The author's demonstrated that by injecting even just 50 samples, they were able to obtain a success rate above 90% in the different strat-egies employed, and even 100% in some of them. The authors also demonstrated the existence of physical backdoors, wherein a physical accessory could act as the backdoor for a visual deep network. They were able to achieve 100% success rates with 20 poisoning samples, and a 0% success rate for wrong keys.

B. Attacks using Deep Neural Networks

Due to their potential for misuse, the offensive applications of AI have been limited to research-based papers only. However, these experiments show the capability of using AI and Deep Learning as an offensive tool for causing harm. The following paragraphs cover the malicious use of deep learning to affect digital security.

Bahnsen et al (2018) [66] used LSTM networks to bypass intrusion detection systems. They analyzed more than a million phishing URLs to understand the different strategies that threat actors use to create phishing URLs. Using LSTMs on these URL strings, they created the so-called DeepPhish, an algorithm that learns to create better phishing attacks by learning and replicating the inner structure of effective malicious URLs. By training the DeepPhish algorithm for two different threat actors, they were able to increase their effectiveness (per-centage of attacks not blocked by a proactive (real-time) phishing detection system) from 0.69% to 20.9%, and 4.91% to 36.28%, respectively.

Hitaj et al (2019) [67] created PassGAN, a generative adversarial network used for password guessing. It learns the distribution of real passwords from actual password leaks and generates high-quality password guesses. They evaluated PassGAN on LinkedIn and RockYou password datasets, and were able to surpass rule-based and state-of-the-art machine learning password guessing tools at accuracies of 34.6% and 34.2% respectively, when the training and test data pass-words were mutually exclusive. PassGAN did not require a-priori knowledge on passwords or common password structures, and was able to learn intrinsic features autonomously. In addition, the authors combined the output of PassGAN with the output of HashCat, a state-of-the-art password guessing tool, to match 51%-73% more passwords than with HashCat alone.

Seymour and Tully (2017) [68] demonstrated that a fully automated spear phishing system could create tailored tweets on the social media platform Twitter based on the individual user's interests, possibly directing the user towards potentially malicious links. They developed an RNN named SNAP_R, which was trained using a combination of spear phishing pen-testing data, Reddit submissions and Twitter tweets. The model was seeded dynamically to ensure that tweets were generated on trending topics that were more likely to be clicked. Their tests on 90 users revealed a success rate between 30% and 66% which is comparable to large-scale manual spear phishing efforts.

Anderson et al. (2017) [69] created a machine learning model, DeepDGA, to automatically generate command and control domains that are indistinguishable from legitimate domains by human and machine observers. They develop a GAN to generate domain names that are able to evade DGA detection classifier that is also trained by the authors using random forest algorithm. The random forest classifiers achieved a minimum accuracy of 85% on the previous datasets, which dropped to 48% on DeepDGA generated samples. Finally, they used DeepDGA to create an augmented dataset with adversarial examples, and used this to train the classifier. This method improved the average accuracy over different DGA families, from 68% to 70%, while the individual accuracies improved or were at par with the baseline in all cases but two.

Side channel attacks exploit the information gained from the physical characteristics of a computer, such as timing, power consumption, voltage peaks etc., to deduce private information from the system. Maghrebi et al (2016) [70] and Yu et al (2018) [71] explore the use of deep learning in assisting side channel at-tacks. Maghrebi et al compared machine learning

and deep learning techniques for the purpose of side channel key recovery in template attacks [72]. The authors tested the techniques on different implementations of AES encryption algorithm and found that deep learning algorithms outperformed state-of-the-art techniques. They concluded that deep learning techniques provided better features that were suitable for key recovery, such as feature extraction in CNNs and time dependency in RNN/LSTMs. Yu et al utilized DNNs for key recovery in AES implementations that use dynamic voltage scaling (DVS). The DNN is trained to estimate power dissipation in the AES circuit, by analyzing its previous emissions. By correlating the power dissipation with encrypted data, and comparing predicted and actual power dissipations, the attackers are able to obtain the secret AES key. The models attained a correlation of 0.8146 and a deviation of 11.63% compared to actual output, on a dataset of 3x104 samples, which was deemed adequate for effective performance by the authors.

Sivakorn et al (2016) [73] studied reCaptcha, a security system used to detect whether a computer user on a website is human or a bot. They identify flaws in the system that could allow adversaries to bypass the security check and perform malicious activities on the network. They designed a novel, low-cost attack. Their solution consisted of three modules for image annotation, identification of the tag (hint) of the image and for referencing previous challenges in case of repeated images. The authors used pre-trained and publicly available deep learning frameworks for detecting image annotations. They achieved an accuracy of 70.78% with image based reCaptcha, and required computational time of 19 seconds. They also tested the strategy with Facebook captcha, and achieved maxi-mum accuracy of 83.5% when using Clarifai framework for image annotation. The authors concluded with the suggestion to study and improve the current state of captchas.

Spectre (2019) [74] is an exploit of two CPU optimizations: branch prediction and speculative execution. Speculative execution copies RAM read by the CPU into cache, in case it is required in the future. However, the processors are not able to determine whether the RAM addresses copied were authorized for copying. This flaw can be exploited to trick the CPU into caching forbidden memory which can be read by a cache side channel timing attack. This attack exploits timing differences in how long a CPU takes to fetch values from the RAM vs an embedded CPU cache. However, every CPU will exhibit different timing behavior. Therefore, a deep learning model [75] can be used to learn the cache timings of the CPU. The model takes as input the all possible cache timing values for a single byte, and outputs the actual byte value, based on the output neuron with highest probability.

VI. CONCLUSION AND FUTURE SCOPE

In this paper, we first presented the various applications of deep learning techniques in the field of cybersecurity. In addition, we showed the security issued of using deep learning: how their applications can be exploited and how they can be used for malicious activities. The paper aimed to demonstrate the role deep learning plays in cybersecurity, and how deep learning solutions can perform better than, or already are, state-of-the-art. We suggest that researchers and organizations can consider incorporating deep learning into security infrastructures. Moreover, more research needs to be done in order to improve the security of neural networks, as they are susceptible to adversarial and poisoning attacks. Finally, with new arising technologies, it is imperative that we create certain standards for the use of AI as stated in [5], in order to prevent unethical and illegal use of these algorithms. By employing these standards and preparing defenses against deep learning attacks, deep learning applications can be integrated into many cyber-security areas with great success.

Future work in this area can focus on the new and rising technologies such as 5G, IoT, blockchain, quantum computing and edge computing, and their effect on deep learning. These new technologies will come with their own security issues. Also, new and complex deep learning architectures that will be developed in the future would also need to be evaluated. We hope that this work motivates further exploration into the applications of deep learning in cybersecurity.

VII. REFERENCE

[1] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural net-works, 61, 85-117.

[2] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[3] Dilek, S., Çakir, H., & Aydin, M. (2015). Applications of Artificial Intelligence Tech-niques to Combating Cyber Crimes: A Review. International Journal of Artificial Intelli-gence & Applications, 6(1), 21.

[4] Berman, D. S., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2019). A survey of deep learning methods for cyber security. Information, 10(4), 122.

[5] Brundage, M., Avin, S., Clark, J., Toner, H., Eckersley, P., Garfinkel, B., ... & Anderson, H. (2018). The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mit-igation. arXiv preprint arXiv:1802.07228.

[6] Apruzzese, G., Colajanni, M., Ferretti, L., Guido, A., & Marchetti, M. (2018, May). On the effectiveness of machine and deep learning for cyber security. In 2018 10th Interna-tional Conference on Cyber Conflict (CyCon) (pp. 371-390). IEEE.

[7] Bae, H., Jang, J., Jung, D., Jang, H., Ha, H., & Yoon, S. (2018). Security and Privacy Is-sues in Deep Learning. arXiv preprint arXiv:1807.11655.

[8] Akhtar, N., & Mian, A. (2018). Threat of adversarial attacks on deep learning in comput-er vision: A survey. IEEE Access, 6, 14410-14430.

[9] Zell, A. (1994). Simulation neuronaler netze (Vol. 1, No. 5.3). Bonn: Addison-Wesley.

[10] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). IEEE.

[11] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recogni-tion. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[12] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the in-ception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).

[13] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information pro-cessing systems (pp. 2672-2680).

[14] Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[15] Lu, Y., Wu, S., Tai, Y. W., & Tang, C. K. (2018). Image generation from sketch con-straint using contextual GAN. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 205-220).

[16] Larsen, A. B. L., Sønderby, S. K., Larochelle, H., & Winther, O. (2016, June). Autoen-coding beyond pixels using a learned similarity metric. In International Conference on Machine Learning (pp. 1558-1566).

[17] Siegelmann, H. T., & Sontag, E. D. (1992, July). On the computational power of neural nets. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 440-449). ACM.

[18] Cannady, J. (1998, October). Artificial neural networks for misuse detection. In National information systems security conference (Vol. 26).

[19] Wu, C. H. (2009). Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks. Expert Systems with Applications, 36(3), 4321-4330.

[20] Salvador, P., Nogueira, A., Franca, U., & Valadas, R. (2009, May). Framework for zom-bie detection using neural networks. In 2009 Fourth International Conference on Internet Monitoring and Protection (pp. 14-20). IEEE.

[21] Tuor, A., Kaplan, S., Hutchinson, B., Nichols, N., & Robinson, S. (2017, March). Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. In Workshops at the Thirty-First AAAI Conference on Artificial Intelligence.

[22] Bitter, C., Elizondo, D. A., & Watson, T. (2010, July). Application of artificial neural networks and related techniques to intrusion detection. In The 2010 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[23] Depoix, J., & Altmeyer, P. (2018). Detecting Spectre Attacks by identifying Cache Side-Channel Attacks using Machine Learning. Advanced Microkernel Operating Systems, 75.

[24] R, Vinayakumar, HB, B. G., Poornachandran, P., & KP, S. (2018). Deep-Net: Deep Neu-ral Network for Cyber Security Use Cases. arXiv preprint arXiv:1812.03519.

[25] Yu, B., Gray, D. L., Pan, J., De Cock, M., & Nascimento, A. C. (2017, November). In-line DGA detection with deep networks. In 2017 IEEE International Conference on Data Mining Workshops (ICDMW) (pp. 683-692). IEEE.

[26] Feng, Z., Shuo, C., & Xiaochuan, W. (2017, December). Classification for DGA-based malicious domain names with deep learning architectures. In 2017 Second International Conference on Applied Mathematics and information technology (p. 5).

[27] Hendler, D., Kels, S., & Rubin, A. (2018, May). Detecting malicious PowerShell com-mands using deep neural networks. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security (pp. 187-197). ACM.

[28] Shibahara, T., Yamanishi, K., Takata, Y., Chiba, D., Akiyama, M., Yagi, T., ... & Murata, M. (2017, May). Malicious URL sequence detection using event de-noising convolution-al neural network. In 2017 IEEE International Conference on Communications (ICC) (pp. 1-7). IEEE.

[29] Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017, January). Malware traffic clas-sification using convolutional neural network for representation learning. In 2017 Inter-national Conference on Information Networking (ICOIN) (pp. 712-717). IEEE.

[30] Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2016, December). Deep learning for classification of malware system call sequences. In Australasian Joint Conference on Artificial Intelligence (pp. 137-149). Springer, Cham.

[31] Torres, P., Catania, C., Garcia, S., & Garino, C. G. (2016, June). An analysis of recurrent neural networks for botnet detection behavior. In 2016 IEEE biennial congress of Argen-tina (ARGENCON) (pp. 1-6). IEEE.

[32] McDermott, C. D., Majdani, F., & Petrovski, A. V. (2018, July). Botnet detection in the internet of things using deep learning approaches. In 2018 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[33] Staudemeyer, R. C. (2015). Applying long short-term memory recurrent neural networks to intrusion detection. South African Computer Journal, 56(1), 136-154.

[34] Kim, J., & Kim, H. (2015, August). Applying recurrent neural network to intrusion detec-tion with hessian free optimization. In International Workshop on Information Security Applications (pp. 357-369). Springer, Cham.

[35] Kim, G., Yi, H., Lee, J., Paek, Y., & Yoon, S. (2016). LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection sys-tems. arXiv preprint arXiv:1611.01726.

[36] Loukas, G., Vuong, T., Heartfield, R., Sakellari, G., Yoon, Y., & Gan, D. (2017). Cloud-based cyber-physical intrusion detection for vehicles using deep learning. Ieee Access, 6, 3491-3508.

[37] Maniath, S., Ashok, A., Poornachandran, P., Sujadevi, V. G., Sankar, A. P., & Jan, S. (2017, October). Deep learning LSTM based ransomware detection. In 2017 Recent De-velopments in Control, Automation & Power Engineering (RDCAPE) (pp. 442-446). IEEE.

[38] Ferdowsi, A., & Saad, W. (2018, May). Deep learning-based dynamic watermarking for secure signal authentication in the Internet of Things. In 2018 IEEE International Confer-ence on Communications (ICC) (pp. 1-6). IEEE.

[39] Kong, H., Lu, L., Yu, J., Chen, Y., Kong, L., & Li, M. (2019, July). FingerPass: Finger gesture-based continuous user authentication for smart homes using commodity WiFi. In Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Net-working and Computing (pp. 201-210). ACM.

[40] Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Chandrasekhar, V. R. (2018). Efficient gan-based anomaly detection. arXiv preprint arXiv:1802.06222.

[41] Chen, H., & Jiang, L. (2019). GAN-based method for cyber-intrusion detection. arXiv preprint arXiv:1904.02426.

[42] Volkhonskiy, D., Nazarov, I., Borisenko, B., & Burnaev, E. (2017). Steganographic gen-erative adversarial networks. arXiv preprint arXiv:1703.05502.

[43] Shi, H., Dong, J., Wang, W., Qian, Y., & Zhang, X. (2017, September). SSGAN: secure steganography based on generative adversarial networks. In Pacific Rim Conference on Multimedia (pp. 534-544). Springer, Cham.

[44] Lee, H., Han, S., & Lee, J. (2017). Generative adversarial trainer: Defense to adversarial perturbations with gan. arXiv preprint arXiv:1705.03387.

[45] Hayes, J., Melis, L., Danezis, G., & De Cristofaro, E. (2017). LOGAN: evaluating priva-cy leakage of generative models using generative adversarial networks. arXiv preprint arXiv:1705.07663.

[46] Yin, C., Zhu, Y., Liu, S., Fei, J., & Zhang, H. (2018, May). An enhancing framework for botnet detection using generative adversarial networks. In 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD) (pp. 228-234). IEEE.

[47] Wang, X., & Yiu, S. M. (2016). A multi-task learning model for malware classification with useful file access pattern from API call sequence. arXiv preprint arXiv:1610.05945.

[48] Lotfollahi, M., Siavoshani, M. J., Zade, R. S. H., & Saberian, M. (2017). Deep packet: A novel approach for encrypted traffic classification using deep learning. Soft Computing, 1-14.

[49] Aminanto, M. E., & Kim, K. (2017, August). Improving detection of Wi-Fi impersona-tion by fully unsupervised deep learning. In International Workshop on Information Se-curity Applications (pp. 212-223). Springer, Cham.

[50] Kolias, C., Kambourakis, G., Stavrou, A., & Gritzalis, S. (2015). Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. IEEE Communica-tions Surveys & Tutorials, 18(1), 184-208.

[51] Sun, J., Wang, X., Xiong, N., & Shao, J. (2018). Learning sparse representation with var-iational auto-encoder for anomaly detection. IEEE Access, 6, 33353-33361.

[52] Ma, T., Wang, F., Cheng, J., Yu, Y., & Chen, X. (2016). A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks. Sen-sors, 16(10), 1701.

[53] Abeshu, A., & Chilamkurti, N. (2018). Deep learning: the frontier for distributed attack detection in fog-to-things computing. IEEE Communications Magazine, 56(2), 169-175.

[54] Alam, M., Bhattacharya, S., Dutta, S., Sinha, S., Mukhopadhyay, D., & Chattopadhyay, A. (2019, May). RATAFIA: Ransomware Analysis using Time And Frequency Informed Autoencoders. In 2019 IEEE International Symposium on Hardware Oriented Security and Trust (HOST) (pp. 218-227).

[55] Shi, C., Liu, J., Liu, H., & Chen, Y. (2017, July). Smart user authentication through actu-ation of daily activities leveraging WiFi-enabled IoT. In Proceedings of the 18th ACM In-ternational Symposium on Mobile Ad Hoc Networking and Computing (p. 5). ACM.

[56] Tabacof, P., Tavares, J., & Valle, E. (2016). Adversarial images for variational autoen-coders. arXiv preprint arXiv:1612.00155.

[57] Kos, J., Fischer, I., & Song, D. (2018, May). Adversarial examples for generative mod-els. In 2018 IEEE Security and Privacy Workshops (SPW) (pp. 36-42). IEEE.

[58] Rozsa, A., Günther, M., Rudd, E. M., & Boult, T. E. (2019). Facial attributes: Accuracy and adversarial robustness. Pattern Recognition Letters, 124, 100-108.

[59] Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In Proceedings of the IEEE international conference on computer vision (pp. 3730-3738).

[60] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017, April). Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia conference on computer and communications security (pp. 506-519). ACM.

[61] Ilyas, A., Engstrom, L., Athalye, A., & Lin, J. (2018). Black-box adversarial attacks with limited queries and information. arXiv preprint arXiv:1804.08598.

[62] Athalye, A., Engstrom, L., Ilyas, A., & Kwok, K. (2017). Synthesizing robust adversarial examples. arXiv preprint arXiv:1707.07397.

[63] Yang, C., Wu, Q., Li, H., & Chen, Y. (2017). Generative poisoning attack method against neural networks. arXiv preprint arXiv:1703.01340.

[64] Muñoz-González, L., Biggio, B., Demontis, A., Paudice, A., Wongrassamee, V., Lupu, E. C., & Roli, F. (2017, November). Towards poisoning of deep learning algorithms with back-gradient optimization. In Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (pp. 27-38). ACM.

[65] Chen, X., Liu, C., Li, B., Lu, K., & Song, D. (2017). Targeted backdoor attacks on deep learning systems using data poisoning. arXiv preprint arXiv:1712.05526.

[66] Bahnsen, A. C., Torroledo, I., Camacho, L. D., & Villegas, S. (2018, May). DeepPhish: Simulating Malicious AI. In Proceedings of the Symposium on Electronic Crime Re-search, San Diego, CA, USA (pp. 15-17).

[67] Hitaj, B., Gasti, P., Ateniese, G., & Perez-Cruz, F. (2019, June). Passgan: A deep learning approach for password guessing. In International Conference on Applied Cryptography and Network Security (pp. 217-237). Springer, Cham.

[68] Seymour, J., & Tully, P. (2016). Weaponizing data science for social engineering: Auto-mated E2E spear phishing on Twitter. Black Hat USA, 37.

[69] Anderson, H. S., Woodbridge, J., & Filar, B. (2016, October). DeepDGA: Adversarially-tuned domain generation and detection. In Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security (pp. 13-21). ACM.

[70] Maghrebi, H., Portigliatti, T., & Prouff, E. (2016, December). Breaking cryptographic implementations using deep learning techniques. In International Conference on Securi-ty, Privacy, and Applied Cryptography Engineering (pp. 3-26). Springer, Cham.

[71] Yu, W., & Chen, J. (2018). Deep learning-assisted and combined attack: a novel side-channel attack. Electronics Letters, 54(19), 1114-1116.

[72] Chari, S., Rao, J. R., & Rohatgi, P. (2002, August). Template attacks. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 13-28). Springer, Berlin, Heidelberg.

[73] Sivakorn, S., Polakis, J., & Keromytis, A. D. (2016). I'm not a human: Breaking the Google reCAPTCHA. Black Hat.

[74] Kocher, P., Horn, J., Fogh, A., Genkin, D., Gruss, D., Haas, W., ... & Schwarz, M. (2019, May). Spectre attacks: Exploiting speculative execution. In 2019 IEEE Symposium on Security and Privacy (SP) (pp. 1-19). IEEE.

[75] Exploiting Spectre with Deep Learning, https://medium.com/hackernoon/exploiting-spectre-with-deep-learning-d8ec2ba4c8ca, last accessed 2019/12/09.