sustainability

MDPI

Article

Juice Jacking: Security Issues and Improvements in USB Technology

Debabrata Singh [1,†], Anil Kumar Biswal [2,†], Debabrata Samanta [3,†], Dilbag Singh [4,†] and Heung-No Lee [4,*,†]

1 Department of Computer Application, Institute of Technical Education and Research (ITER), Siksha 'O'Anusandhan (SOA) Deemed to be University, Bhubaneswar 751030, Odisha, India; debabratasingh@soa.ac.in
2 Department of Computer Science and Engineering, Institute of Technical Education and Research (ITER), Siksha 'O'Anusandhan (SOA) Deemed to be University, Bhubaneswar 751030, Odisha, India; anil.biswal123@gmail.com
3 Department of Computer Science, CHRIST University, Bangalore 560029, Karnataka, India; debabrata.samanta369@gmail.com
4 School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology (GIST), 261 Cheomdan-Gwagiro (Oryong-dong), Buk-gu, Gwangju 61005, Korea; dggill2@gmail.com
* Correspondence: heungno@gist.ac.kr
† These authors contributed equally to this work.

Abstract: For a reliable and convenient system, it is essential to build a secure system that will be protected from outer attacks and also serve the purpose of keeping the inner data safe from intruders. A juice jacking is a popular and spreading cyber-attack that allows intruders to get inside the system through the web and theive potential data from the system. For peripheral communications, Universal Serial Bus (USB) is the most commonly used standard in 5G generation computer systems. USB is not only used for communication, but also to charge gadgets. However, the transferal of data between devices using USB is prone to various security threats. It is necessary to maintain the confidentiality and sensitivity of data on the bus line to maintain integrity. Therefore, in this paper, a juice jacking attack is analyzed, using the maximum possible means through which a system can be affected using USB. Ten different malware attacks are used for experimental purposes. Various machine learning and deep learning models are used to predict malware attacks. An extensive experimental analysis reveals that the deep learning model can efficiently recognize the juice jacking attack. Finally, various techniques are discussed that can either prevent or avoid juice jacking attacks.

Keywords: cyber-attack; malicious code; USB code; security; hacker; keystroke dynamics; authentication

Citation: Singh, D.; Biswal, A.K.; Samanta, D.; Singh, D.; Lee, H.-N. Juice Jacking: Security Issues and Improvements in USB Technology. *Sustainability* **2022**, *14*, 939. https://doi.org/10.3390/su14020939

Academic Editors: Marko Hölbl, SK Hafizul Islam, Marimuthu Karuppiah and Chien-Ming Chen

Received: 30 November 2021
Accepted: 11 January 2022
Published: 14 January 2022

1. Introduction

Juice jacking is a well-known cyber-attack used to attack Universal Serial Bus (USB)-enabled devices such as mobiles, tablets, and laptops. It generally utilizes the charging port of a given device; then, whenever someone connects a given device to the system using this port, the hackers obtain all their personal information or may upload some malware onto the device. Therefore, it is necessary to detect and prevent these kinds of attacks.

Business travelers now have access to public USB power charging stations at airports, hotels, and other places to which they travel or stay [1]. In Android OS and iOS, this attack is more appropriate and the smartphone's display can be exposed through a standard Micro-USB [2,3] connected using the Mobile High-Definition Link (MHL) standard or the iPhone's lightning connector.

However, even without the instant injection of malware, a leakage to a rogue kiosk might cause a continual security threat [4]. An address book, images, music, and SMS are just a few of the items that may be accessible once the device is associated with a computer. Both data and power transfer can be accomplished using USB connectors [2,5].

A decade ago, security researchers worked out how to exploit USB connections, which a user might think are solely used to transfer power, to hide and transport secret data payloads, as cellphones grew more prevalent [6,7].

Markus, President of Aries security, and his fellow researchers Joseph Mlodzianowski and Robert Rowley, built the charging kiosk. They made charging stations more attractive with a variety of charging cables. When no device is connected, the charging station displays a blue image with the words "Free cell phone charging kiosk" and whenever any device is connected a red warning sign shows and a message: "Be careful and should not trust public kiosks" [8,9]. The wall of sheep is an event held at Defcon, which has allowed public access to juice jacking kiosks every year since 2011. This can raise awareness among the public [8,10]. In addition, juice jacking can be used by hackers to inject malicious code onto the devices and obtain information on those devices.

Regarding device connectivity, the Internet of Things (IoT) has offered the world a higher level of accessibility, integrity, availability, scalability, secrecy, and interoperability. IoTs, on the other hand, are vulnerable to security threats due to a mixture of various attack surfaces and their newness, resulting in a lack of security standardization and criteria. Attackers can use a wide range of cyberattacks on IoTs, depending on which aspect of the system they are targeting and what they expect to achieve from the attack [11]. As a result, a significant amount of research has been dedicated to building secure IoT devices. Recently, Artificial Intelligence (AI)-enabled approaches have been extensively utilized to implement secure IoT devices and networks. Typically, AI models identify anomalous activity, which helps to predict a given attack. In IoTs, cyber-attackers always have an advantage because they only need to uncover one vulnerability, whereas cybersecurity specialists must secure several targets [12]. Recently, various supervised learning models, such as decision trees, linear regression, machine learning, support vector machines, and neural networks, have been employed in IoT cybersecurity applications to predict threats.

1.1. Types of Juice Jacking

- Data theft: In data theft, cybercriminals steal all information from the device, i.e., devices connected to charging stations through USB ports. As a result, hackers drop an additional payload to steal the information from the connected device [13].
- Malware installation: Malware is loaded on the linked device and remains there until it is recognized and uninstalled by the user. Cybercriminals use malware such as adware, ransomware, and Trojans [14,15].
- Countermeasures: The best approach to avoid juice jacking attacks is to stay away from portable wall chargers and public charging stations [16]. You should keep an external battery or power bank. Random AC outlets have fewer risks than public USB stations. If there is no solution other than using a public charging station, then adapters are available in the device to block data transfer during charging.

Numerous methods are used to prevent juice jacking. These include ensuring devices are charged, avoiding the use of USB chargers, turning off gadgets while not in use, and purchasing charging-only cables. Phone security features and data blocks can also be used.Certain means and softwares can inform you if your phone is hacked, including battery drainage, poor performance, high data usage, and mysterious pop-ups. USB hardware can be divided into three types: programmable microcontrollers, USB peripherals (maliciously reprogrammed peripherals and non-reprogrammed peripherals), and electrical.

1.2. Features of Juice Jacking Attacks/Malware

The major features of juice jacking attacks/malware attacks are discussed as follows:

- Easy to implement but quite adequate.
- No need to install any more factors on phones, as the attacker does not require the installation of any additional software.

- Does not need to ask for permission, as the attacker does not need to ask for permission from the user or install any apps on the phone.
- Less user conjecture: the user is less aware of charging attacks than malware attacks.
- Multi-platform: the attack is possible in androids as well as iPhones.

1.3. Motivation

As juice jacking is a software-based threat, it requires an acknowledgment that the software is fixed on the device and applicable on a limited platform, that is, Android OS and iOS. Therefore, it is better to avoid hardware-based vulnerabilities such as charging attacks by not installing too much software/security on a device. When a device is in charging mode, a juice jacking attack can automatically record the device's screen and manually extract specific information [17,18]. Since devices such as mobiles, tablets, and notebooks contain confidential and sensitive data; therefore, it is necessary to secure these electronic devices against various attacks such as juice jacking. The main objective of this paper is to analyze juice jacking attacks by considering the maximum possible ways through which a system can be affected using USB. In addition, various techniques will be discussed, which can either be used to prevent or avoid the juice jacking attack.

1.4. Contributions

Juice jacking is a widespread cyber-attack that allows attackers to hack the given system using USB to steal the system's data. Thus, USB is no longer a simple mechanism, used to transfer data or charge the devices due to security concerns. The main contributions of this paper are as follows:

1. Juice jacking attack is analyzed with the maximum possible ways through which a system can be affected using USB.
2. Ten different malware attacks are used for experimental purposes.
3. Various machine learning and deep learning models are used to predict the malware attacks.
4. Finally, various techniques are also discussed, which can either prevent or avoid juice jacking attacks.

The remaining paper can be organized as follows. Section 2 presents the related work on the juice jacking attack. Section 3 categorizes various USB attacks along with various security challenges. The working principle of the juice jacking attack is discussed in Section 4. Performance analyses are presented in Section 5. Various techniques that can be used to prevent juice jacking attacks are discussed in Section 6. Section 7 concludes the paper, along with the future scope.

2. Related Work

Noyes et al. [19] described USB as a peripheral communication in the IT industry. Due to hot plugging and hot swapping, USB is a dominant choice. The ubiquitous USB is a type of USB device, which has good security, but at present, there is a huge gap in USB security, leading to data being hacked. The aim was to analyze susceptibility of the USB code so that it can capture the input on the bus line and provide a care mechanism that will save the USB from active as well as passive attacks. Additional security was available, that can encrypt and verify the USB input. Nissim et al. [20] discussed how attackers are increasing day by day, and how attackers take advantage of users using USB peripheral devices with an embedded malicious payload. Various attacks on persons and organizations have occurred recently, and a new classification has been developed, with the objective of identifying the associations and vulnerable USB peripherals. He [21] argued for an interaction between the computer USB port and portable accessory and proposed a solution based on the analysis results and circuit test.

According to the USB 2.0 power delivery standard, the maximum supply current is 0.5 A at 5 V. USB 3.0 has the maximum amount stream: 0.9 A at 5 V. However, some portable devices can be 2.5 A or more, depending on the design. Anderson et al. [22]

described the advantages of USB flash drives and the favorable approach of relief, and also examines the distinct category of malware and reproduction vectors. Once a proper understanding of the logic has been developed, users can protect their data. Meng Weizhi et al. [23] described Smartphone malware as a popular issue. As public charging is a big concern, users will have a good understanding of their privacy, it has been shown that the normal micro-USB connector, MHL standard, or an iPhone's lighting connection can be used to expose Android OS and iOS displays. Mei-Hong et al. [24] showed the various USB key-based means of software security including honesty evidence, approach authority, and closed cache. Honesty evidence was used to assure flexibility in a USB key and mutual verification was achieved among software and the USB key to check the effectiveness, change password technology and avoid a fake attack. Encrypted content was used to hide information and MAC to assure that the connection content was destroyed.

Lau et al. [25] designed a MACTANS technique for Apple iOS devices to investigate the level of security threats. How an iOS device can be compromised within one minute of being plugged into a rogue charger was demonstrated. Apple's existing security features for the prevention of unauthorized software installation were also discussed. Thereafter, how USB abilities can be used to circumvent iOS defenses was also exhibited, as well as how an attacker can hide their malware by using built-in features of Apple devices. Spolaor et al. [26] presented a no-free-charge theorem. However, while phone recharging is free, there is no guarantee that public charging stations are not intentionally controlled by an adversary. It was shown that the adversary could use a maliciously controlled charging station to exfiltrate data from electronic devices using USB charging cables. A simple application was used, which does not require any permission before data are sent out of the device.

Related work show that the majority of researchers have not utilized deep learning techniques for malware classification. Recently, deep learning techniques [27–29] have received attention from many researchers due to their wide range of applications. Unlike the machine learning techniques, deep learning models do not require hand-crafted features [30, 31]. In addition, deep learning models are the least affected by the over-fitting problem [27, 32].

3. USB Attacks

This section discusses various categories of USB attacks and various security challenges in USB technology.

3.1. Categories of USB Attacks

There are many types of USB attacks, but some commonly used attacks are programmable microcontrollers, a bad USB, and a rubber ducky. The programmable microcontrollers form a well-known USB attack. Human interface devices (HID) such as keyboards and mice are used by USB microprocessors to execute keystroke commands on a target gadget. A bad USB can inexpensively cause a worrying attack. This attack exploits a flaw in the USB firmware. It reprograms the USB device so that it functions as a human interface. When a victim's computer is connected to a USB device, it can be used to execute commands or launch a malicious program. Rubber Ducky injects keystrokes at a rapid speed, violating the inherent trust computers have in humans by posing as a keyboard. It also saves a lot of effort by targeting susceptible systems or programming processes. Payloads can be written using a basic scripting language and online payload generators are also available [33].

3.2. Security Challenges in USB Technology

Retaining the confidentiality, sensitivity, and isolation of a bus line, how do these USB devices maintain their integrity? How does a USB line or traffic send data to a particular malicious node or an organization? To overcome the above problems, a software overlay can be used for encryption on used devices. For secure communication, these

three techniques mainly focus on the validity of the information, data confidentiality, and preventing other devices and different attacks [34].

As the data and information transmitted through the plain text and encryption techniques are applied to secure the bus line [35–37], both the USB device and the host controller built a secure channel to improve the USB line using the concept of encryption. A secure socket layer (SSL) can be used for secure communication through the Internet. Through an encrypted channel on a bus line, both the host and the device can initiate a key exchange. As long as both devices use the same cipher, they can exchange data. Private as well as public keys are chosen to decrypt the data, encryption setup, and key exchange concepts, which are depicted in Figure 1.

To ensure that data integrity is maintained, the USB needs to be improved. For secure communication, some type of verification and authentication is required to prevent data from being exploited or tampered with in any way [38]. Digital signatures and public-key cryptography can be used to validate and optimize firmware queries. As seen in this document, sections are numbered using uppercase Arabic numerals, followed by uppercase Arabic numerals that are separated by periods [39,40]. It is not necessary to indent the first paragraphs after the section title. It is only in the first paragraph that a drop cap is used.

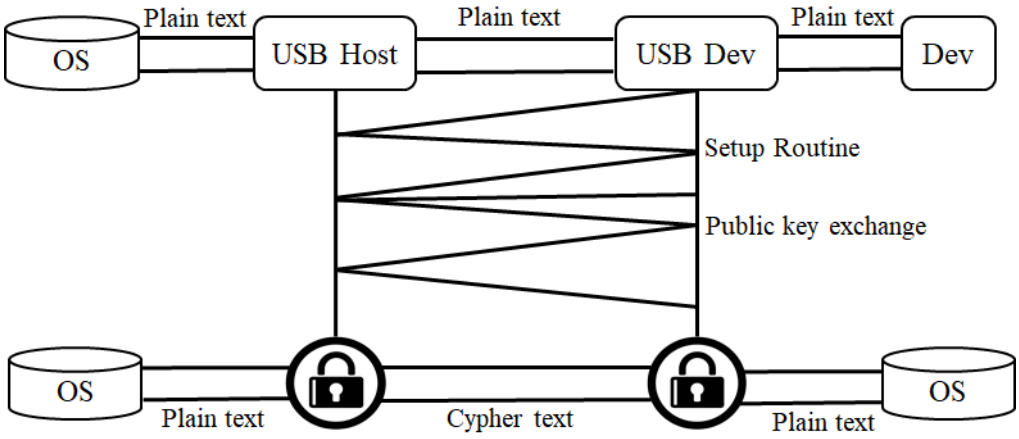

Figure 1. Establish Encryption on USB.

3.3. Designed Juice Filming Attacks Major Features

- Simple but quite productive.
- User unawareness.
- Does not need to install any apps on the device side.
- Does not need to ask for any acceptance.
- Cannot be exposed by any current anti-malware software.
- Can be extensible and efficient in both Android OS and iOS.

4. Working Principle of Juice Jacking Attack

Juice jacking works via pairing. For example, if someone connects an iPad and iPhones to a laptop and there is a trusted relationship between them, tools such as XCodes and iTunes can be used to access host details such as music, SMS, photos, videos, and notes. A laptop can even initiate a computer backup of the phone's database; this is exactly how juice jacking works. However, the same USB cable can transfer power as well as data [41,42]. A USB cable provides four pathways: two power conductors and two twisted signal conductors [43,44]. A formal USB connection has five pins: one is used to charge the receiving end, while two of the others are used by default for data transfer unless any changes are made to the setting, and the data transmission is infirm by default.

The network is only noticeable at the end that provides the power, which, in the case of juice jacking, is not the appliance owner. This means that as users attach a USB port to a

charger, they open their pathway to move data, which leads to juice jacking. As a result, there is the threat of actors stealing data or installing malware on the device [45,46]. The twisted-pair D+ and D- conductor has a USB device with full-speed bandwidth devices and data are transferred through D+ and D-, reducing noise and cross-talk, while VBus and Gnd connectors provide power to the USB device [47]. DCP can be found on a USB power adaptor, running attached devices and the battery, whereas CDP is used for charging ports on the host. On the other hand, there are many intimate file icons, such as Microsoft windows icons for videos and pictures, which are used to trap innocent victims into thinking that they are executed file and harmless images and videos, leading them to manually execute the malware files, so that their devices are affected.

Mike Grover, a security researcher, was recently alerted to the same dangers. He built a USB-to-lighting cable that looked like an ordinary cable and had a Wi-Fi chip fitted within, as depicted in Figure 2. This work looks like any other Human Interface Device (HID). Table 1 show a USB colour cables with their descriptions.

Table 1. USB Colour Cables with their Descriptions.

Pins	Name	Cable Color	Description
1.	VBUS	Red	+5v
2.	D-	Blue	Input−
3.	D+	Green	Input+
4.	ID	N/A	Host connected to the signal
5.	GND	Yellow	Ground or not connected.

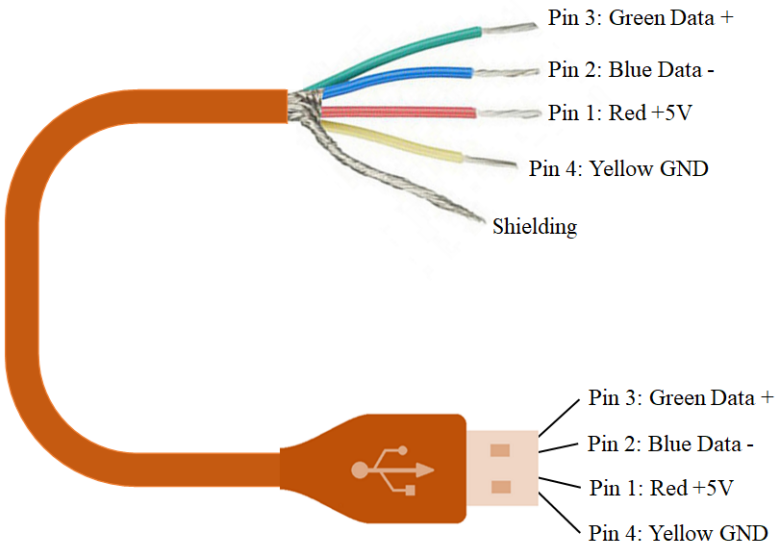

Figure 2. USB pin out wiring

A malicious USB cable or charger can be connected to the cell phone's Wi-Fi or hotspot to execute commands, as shown in Figure 3. There is also an offensive-MG (OMG) cable that can inject malicious data on any device using a smartphone. When buying fake USB cables, users should be cautious because OMG cables can initiate de-authentication assaults on 802.11 and hack a victim's password. Additionally, the operating system should be updated [22].

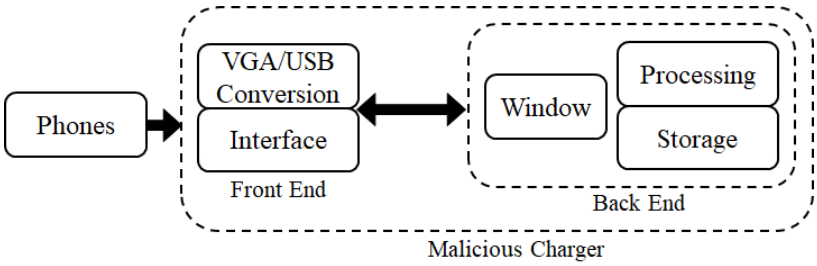

Figure 3. High-level Juice Jacking Charging Attack.

4.1. Working Principle of Juice Filming Charger

The architecture of the juice filming charging (JFC) attack is based on the consideration that no acceptance would be asked for when we plug android or iPhones into a projector; then, the projector can automatically display the phone screen and does not show any bulletin for a plug-in device. VGA/USB allows the attacker to capture user inputs such as password, PIN code, or email address. Displays can be exposed via a regular micro-USB connector that supports the mobile high-definition link (MHL) standard and, for iPhones, a lighting connector is used instead of the micro-USB connector. Figure 4: When a user connects their phone to a JFC charger, the phone's screen can be used to collect the video into several video files at the back-end while this procedure extracts private information from the user. A red, green, blue (RGB) VGA frame-grabber is responsible for the conversion of a video signal from VGA to USB.

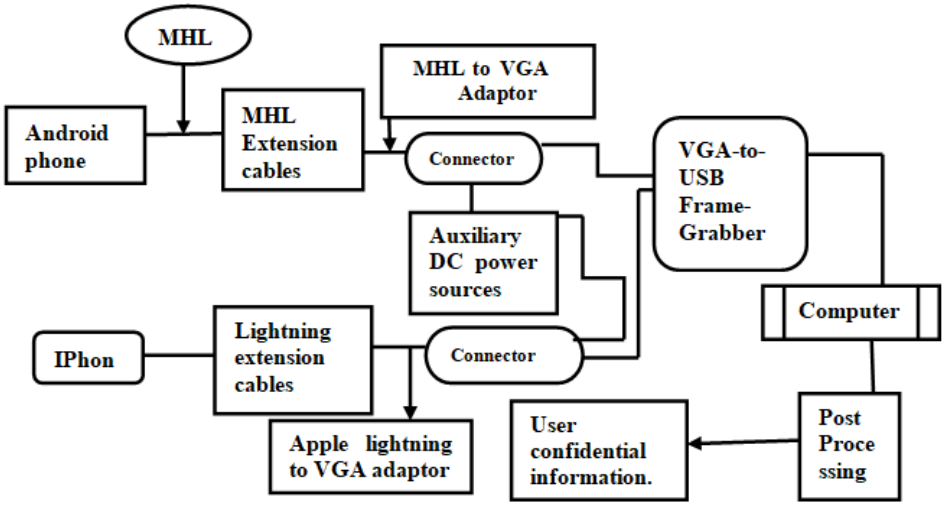

Figure 4. Real setup of Juice Jacking filming charging attack Using VGA2USB.

To evaluate the existence of a juice jacking attack, a deep learning-based model was designed. For detection, Euclidean distance is calculated for the comparison between current usage change and the pre-computed array as follows:

$$[D(P1, P2)]^2 = \sum_{0}^{N} (P1_i - P2_i)^2 \quad (1)$$

Here, D shows distance. $P1_i$ and $P2_i$ define values of the ith attribute of points $P1$ and $P2$, respectively.

Classification accuracy (CA) and hit rate (HR) of normal charger and juice jacking charger can be calculated as:

$$\mathrm{CA} = \frac{(\mathrm{TP} + \mathrm{TN})}{(\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP})} \quad (2)$$

$$\mathrm{HR} = \frac{(\mathrm{TP})}{(\mathrm{TP} + \mathrm{FP})} \tag{3}$$

Here, TP and TN show true positive instances and true negative instances, respectively. FN and FP define false-negative instances and false-positive instances, respectively.

After every 0.05 s, the software received a block of audio, and root means squared error $(RMSE)$ was calculated as:

$$RMSE = \sqrt{\frac{\sum_{x=0}^{y} F_x}{y}} \tag{4}$$

where y denotes the number of audio frequencies captured during 0.05 s. F_x denotes the xth frequency recorded. This block can be identified as a peak and the time of the peak measurement is retained if the resulting $RMSE$ is higher than a predetermined threshold. $RMSE$ is used to find the percentage of malware.

The measurement of voltage and current can be used to determine the power consumption of USB devices. USB power lines have a constant voltage of 5 volts. The fluctuates proportionately to the power usage, so measuring only the current (I_c) is adequate. The analog-to-digital converter (ADC) is then used to measure any voltage drop (V_d) after the resistor (R_g), and power consumption (P_c) that can be computed using Ohm's law, as follows:

$$P_c = V_d \times I_c \tag{5}$$

$$I_c = \frac{V_d}{R_g} \tag{6}$$

$$P_c = \frac{5 \times V_d}{R_g} \tag{7}$$

A linear relationship exists between power consumption and voltage drop, as shown by Equation (7). The constants are the resistor's resistance and the five-volt USB VBUS voltage. This begins when the researcher connects the keyboard to the computer and launches the analyzing computer's microphone delay detection module. To do this, he or she positions the keyboard at a predetermined distance from the microphone. To complete one session, the researcher presses a key 35 times. A session's total delay can be computed as follows:

$$SD = \frac{\sum_{x=1}^{k_p} (MP_x - OP_x)}{k_p} \tag{8}$$

where k_p shows key-pressing frequency. The microphone (MP_x) detects the peak event of a keystroke press in milliseconds. The operating system (OP_x) detects the keystroke press in milliseconds.

4.2. Common Attacker Scenarios

A. Public Charging Stations: Many public areas, including airports and subways, already provide charging stations for users. An attacker can exploit this vulnerability by using a public charger that simply provides an interface for users to charge their phones. It is difficult to determine whether the charger is secure or not because only an interface is offered.

B. Semi-Public Charging Stations: In general, this refers to a hotel or a place in which individuals cannot fully monitor or audit the room, or prepare an attacker in advance and then become the living client victims of an attack.

C. Borrowed Charger: As the name suggests, this refers to a hotel or other place where individuals are unable to monitor and audit the room, prepare the attacker in advance or become the living customer victims.

5. Performance Analyses

In android OS and iOS, we observed the unlock pattern and can also capture user accounts as well as password input, and attacks can be recorded using the source code. Algorithm 1 shows the source code used. The proposed system generates alerts for users whenever new devices are inserted into the system.

Algorithm 1: Pseudocode for recording the data patterns.

```
Data: Device
Result: Obtained data patterns
initialization;
#! /bin/bash ;
while the device is connected with charger do
    mkdir –p output ;
    mkdir –p images ;
    now =(date +% s) #filename ;
    /dev./video0 output/ now.mkv #record for 15 sec ;
    mkdir –p images/now #make new directory for this videos frames ;
    sleep ;
end
```

Table 2 shows the status of the number and percentage of malware. Several malwares and different malware behaviors for smart devices are depicted in Figure 5. Each category consists of various types of malwares/attacks. The mentioned malware percentage (%) depicts the influence of the specific type.

Table 2. Status of number and percentage of malware.

Number of Malware	Behaviour/Type of Malware	Percentage (%) of Malware
1	Viruses, Worms, Spyware and other malicious code	66
2	Spam	61
3	Phishing attacks	36
4	Network hacking	24
5	Thefts of mobile devices	21
6	DoS, DDoS attacks	19
7	Thefts of larger hardware	17
8	Corporate espionage	13
9	Targeted attacks	9
10	None	7

The used data for ten types of malware were computed from 1250 participants. The overall analyses were performed on 1250 participants. Initially, we collected various stateless features and set the target class as Malware type. Thereafter, various machine learning and deep learning models were used to build the juice jacking classification model. A tottal of 70% of the dataset was used to train the models. The remaining 10% and 20% fractions were used for validation and testing of the trained models, respectively.

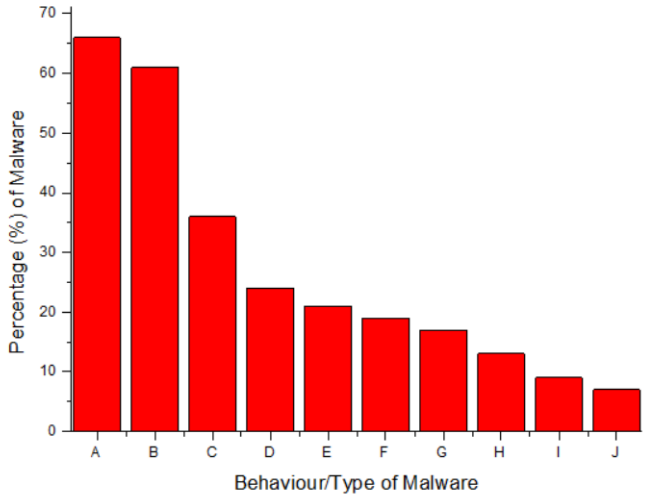

Figure 5. No. of malware and different malware behaviors in Smart Phones.

5.1. Comparative Analysis

In this paper, various machine learning and deep learning models were used to build the juice jacking classification model, such as J48 [48], Support Vector Machine (SVM) [49], k-Nearest Neighbors (KNN) classifier [50], Artificial Neural Networks (ANN) [51], Random Forest (RF) [52], Adaptive Neuro-Fuzzy Inference System (ANFIS) [53,54], and Convolutional Neural Network (CNN) [55,56]. The hyper-parameters of the used models were selected according to their respective default values. Initially, categorical features were converted to numeric features using one-hot encoded vectors. Z-score normalization was then used to normalize the data. To build the deep learning model using CNN, a fully connected layer with 35 size was used. The batch normalization layer was then utilized. Thereafter, a rectified linear unit (ReLU) was considered. Again, a fully connected layer was utilized. Finally, softmax and classification layers were defined. Mini batch size was set as 32. Root Mean Squared Propagation (RMSprop) algorithm was used to optimize the cost function.

Figure 6 shows the confusion matrix analysis of the proposed deep-learning-based juice jacking attack. The deep learning model achieved 76.7% accuracy with a 23.3% error rate.

Table 3 shows a comparative analysis of the proposed model with the competitive models, considering ten malwares. The proposed deep-learning-based model achieved a better performance than the competitive machine learning models. It achieved an average improvement in terms of accuracy, F-score, precision, recall, and hit rate of 2.1789%, 1.9578%, 2.3458%, 2.0696%, and 2.1926%, respectively, compared to the competitive models.

Table 3. Comparative analysis of the proposed system.

Model	Accuracy	F-Score	Precision	Recall	Hit Rate
J48	68.2481	77.9055	70.0374	76.4227	69.7421
SVM	72.4264	79.1583	73.7827	78.8732	72.6881
KNN	70.2702	78.7406	71.9723	77.3199	71.0369
ANN	72.4264	78.1583	73.7827	78.8732	72.6881
RF	72.2433	79.0957	74.8031	79.0391	72.4252
ANFIS	73.0784	78.5149	70.5454	80.9132	73.0316
Deep learning	**76.0784**	**78.1333**	**70.2898**	**82.7683**	**75.3015**

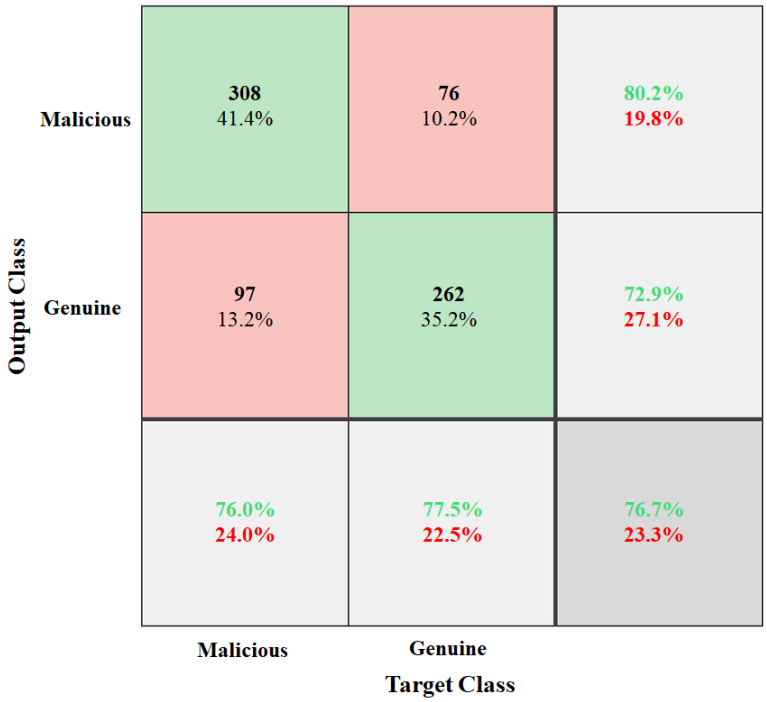

Figure 6. Confusion matrix analysis of the proposed deep-learning-based juice jacking attack.

5.2. Discussion

Table 4 shows a featurewise comparison of this paper with the existing literature. This paper is shown to have considered the impact of juice jacking USB attacks on 1250 participants by considering the ten malware types. We discussed the various methods that are used to prevent juice jacking attacks on both Android OS and iOS. The existing work has mostly considered this attack by looking at specific platforms, such as Andriod or iOS, with a limited number of malware types.

From the proposed work, it is found that when a device is in charging mode, a juice filming attack can automatically record the device's screen and manually extract certain data. While juice jacking is a software-based attack, it necessitates that the device software is acknowledged and fixed. It is also only relevant to devices running Android OS or iOS. Attacks have a higher chance of success when their victims are uninformed or unwilling to pay attention. Automation has increased the potency of our attack. It was also found that the offensive-MG (OMG) cable inserted damaging data on any device when connected to a Wi-Fi network to run commands. Automating the precise charging attack action is a wonderful technique to speed up the process when a high number of films are posted. The proposed system does not require the installation of any additional apps or components on phones, meaning that it is simple to use. A random AC outlet is also used to reduce the risk of using a public USB charger. Finally, we want to draw attention to software such as XCodes and iTunes that can communicate with a laptop and gain access to the host computer.

Table 4. Comparison of the proposed technique with the existing technique.

Approach	Model	Data	Objective	Advantages	Limitations
Proposed Juice Jacking Model	When a device is in charging mode, a juice filming attack can automatically record the device's screen and manually extract specific information. When a high number of films are posted, a manual search can take a lot of time; thus, automating the precise motion of the charging attack is a great way to speed up the process	Ten malware types are tested on over 1250 participants.	To analyze and prevent this attack using the possible ways through which a system can be affected by a USB attack.	The impact of a juice jacking attack is evaluated. Various approaches to avoiding USB-based attacjs are discussed.	Automation of video analysis post-processing and building a malicious recharge station are among the tasks to be completed in the near future.
Mactans (Proof of Concept) Model [25]	Mactans, a malicious charger that can launch malware injection attacks using BeagleBoard after users connect their phones to the charger.	iOS versions up to and including iOS 6.	Protected against various types of attacks when using products such as the iPhone or iPad.	Mactans can connect directly to the Apple Provisioning Portal, enter a target device's UDID, and then acquire a provisioning profile for that device.	Any host that knows the proprietary (SSL) XML RPC-like communications protocol used by iTunes to communicate with an iOS device can similarly and directly query or edit the client's state without the user's authorization.
Sandboxed Android App Model [26]	PowerSnitch is a malicious application that can refer users' data by measuring power consumption over a USB charging cord while charging.	Android OS, PowerSnitch App	We created PowerSnitch, a proof-of-concept programme that can convey data in the form of power bursts by altering the device's CPU power consumption.	USB charging cable was used to steal data from devices connected to a public charging station. A PowerSnitch app was designed that does not require user's permission to be installed on a non-rooted device.	By extending the architecture to incorporate error correction algorithms and synchronisation recover mechanisms, the transmitter and decoder cannot reduce the Bit Error Ration of data transmission.
USB Tracking Model [19]	The document will create an extension to the USB protocol to account for information security, such as data encryption and authentication via the USB line.	Andriod OS and USB protocols.	To investigate the possibility of capturing data on the bus line and demonstrating security mechanics to secure the USB from both passive and active attacks.	USB security was improved by considering the confidentiality, integrity, and availability spectrum in terms of data secrecy, message integrity, and device authentication.	Prototyping security solutions in USB and assessing the tradeoff between security strength and performance efficiency are among the next steps.

6. Security Improvements

This section discusses several ways to keep your device secure and safe.

A. Confidentiality of USB Traffic: On closer inspection, we can see that there is a large amount of data are being delivered and received via a USB network channel.

By performing a "sniffing" attack on the bus, it is possible to read the plain-text addresses of these packets. Sniffing attacks can affect both hardware and software. This proof-of-concept attack can be carried out on the USB bus of a test system. As a result, the USB bus of the system is employed to simulate a keyboard input to a machine. As a result, the user's keystrokes can be exposed. As data are transmitted in plain text, some form of encryption should be utilized to intercept it. To improve the USB line with encryption, a method of establishing a secure channel between the host monitor and the USB device is necessary. Therefore, in this paper, various methods are tested to determine which one would be most effective in protecting communication between these devices [37].

B. The integrity of USB Traffic: The general integrity of data is the next aspect of USB that needs to be addressed. A probabilitys exist that other devices will mistakenly believe that they are separate devices. The use of spoofing can benefit both the system and the end-user [38]. Example: a popular printer that has malicious programming installed on it. As a result of this malicious malware, a keyboard will appear on the machine and orders will be sent. A conceivable attack vector will be substantially widened [13]. It is possible to attach any USB device with this form of attack, which is undetectable until it is used.

C. Authentication of USB: A type of verification and certification is required to secure the bus from tampering and exploitation. For a host to verify that the device is what it was computed to be, a type of device key signature is required. The digital signature approach

can also be used to secure USB data. A public-key cryptography can be used to ensure that each device has a signature that the host can verify and question.

D. Security features of device: Most smart devices have various security but rarely do we use them. When we connect our device to a USB cable, it asks for permission to transfer data, but when we click on cancel our data stop transmitting. The workflow diagram for checking new devices is depicted in Figure 7.

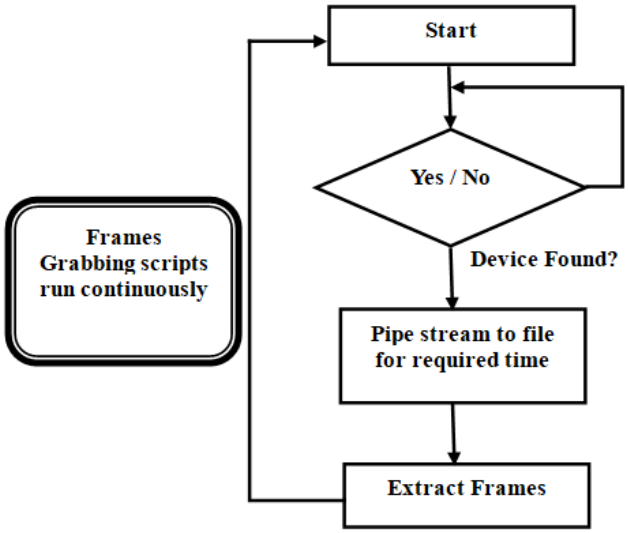

Figure 7. Workflow Diagram of Checking New Devices.

6.1. Prevention of Juice Jacking

- Keep devices charged—whenever we are going out for a long period of time, we should keep our devices charged.
- Avoid USB chargers—public chargers should be avoided and AC/DC chargers should be used, so that there is only a one-way charge for our appliance.
- If a power bank is not available, then you must carry your charger.
- Obtain a charge-only cable. The USB charging-only cables have two conductor cables, and there are four conductor cables with charging ports. Therefore, using charging cables in public places prevents juice jacking.
- Switch off your device and before connecting it to a public charger, then only supply current can be passed to the device.

6.2. Steps to Easily Hack Data from Mobile

To easily hack the data from a mobile, the following steps can be taken:

- First, the mobile is recorded and data can be stolen from the USB device.
- Data are downloaded from the DCIM folder.
- Smartphone offers the necessary performance to attackers or leaks target information.
- If the targeted info is images, then it can easily be hacked without user communication, and if it is a PIN code, then the hacker waits for the user's input.
- Hackers install malicious software or a root device to the user's mobile.

6.3. Precautions

Various precautions can be used to prevent USB attacks, such as:

- Monitoring the appliance for unusual activity.
- Mistrustful apps should be deleted and not installed again.
- In factory settings, the device should be rebuilt.
- Install suitable anti-virus software.
- The system software of mobile devices should be updated regularly.

6.4. Limitations

There are various limitations associated with the deployment of the designed system. These limitations can be categorized as follows:

- **Platform independent:** It is difficult to design platform-independent software that can work on all devices, even they have a different operating system.
- **Permission requirement:** The detection of anti-malware techniques require various key permissions from smartphones. Sometimes, it is not possible to obtain all the system permissions.
- **Complexity:** Some viruses also conduct accelerometer side avenue assaults that need more resources from the phone, for instance, additional power is required to increase user awareness.
- **Deployment:** It is difficult to deploy such software, as system software does not usually allow them.

7. Conclusions and Future Analysis

Juice jacking is a well-known cyber-attack used to attack USB-enabled devices. It generally utilizes the charging port of a given device, and whenever someone connects a given device to the system using this port, then hackers obtain personal information or may upload some malware onto the device. Therefore, it is necessary to detect and prevent these kinds of attacks. Therefore, in this paper, a juice jacking attack was analyzed using the maximum possible ways through which a system can be affected by USB. Ten different malware attacks were used for experimental purposes. The overall analyses were performed on 1250 participants. Initially, various stateless features were collected, and the target class was set as malware type. Thereafter, various machine learning and deep learning models, such as J48, SVM, KNN, ANN, RF, ANFIS, and CNN, were utilized to build the juice jacking classification model. Categorical features were converted to numeric using one-hot encoded vectors. Z-score normalization was then used for normalizing the data. From comparative analyses, the deep learning model was shown to achieve an average improvement over the competitive models in terms of accuracy, F-score, precision, recall, and hit rate, by 2.1789%, 1.9578%, 2.3458%, 2.0696%, and 2.1926%, respectively. Finally, various techniques that can either prevent or avoid juice jacking attacks were also discussed.

In this paper, no new deep learning model was designed for the juice jacking classification model. Therefore, in near future, we will propose a novel model that can achieve more accurate results. Additionally, the hype parameters of the deep learning model were selected on a trial-and-error basis. In the future, metaheuristic techniques could be used to optimize the initial parameters of the deep learning model.

Author Contributions: Conceptualization, D.S. (Debabrata Singh); methodology, A.K.B.; software and validation, formal analysis, and investigation, resources and data curation, D.S. (Debabrata Samanta); visualization and supervision, D.S. (Dilbag Singh) and H.-N.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Research Foundation of Korea (NRF) Grant funded by the Korean government (MSIP) (NRF-2021R1A2B5B03002118) and This research was supported by the Ministry of Science and ICT (MSIT), Korea, under the Information Technology Research Center (ITRC) support program(IITP-2021-0-01835) supervised by the IITP(Institute of Information & Communications Technology Planning & Evaluation.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Waters, D.W. USB Port Controller with Automatic Transmit Retries and Receive Acknowledgements. U.S. Patent 9824045B2, 21 November 2017.
2. Loe, E.L.; Hsiao, H.C.; Kim, T.H.J.; Lee, S.C.; Cheng, S.M. SandUSB: An installation-free sandbox for USB peripherals. In Proceedings of the SandUSB: An Installation-Free Sandbox for USB Peripherals, Reston, VA, USA, 12–14 December 2016; pp. 621–626. doi:10.1109/WF-IoT.2016.7845512.
3. Tran, M.Q.; Elsisi, M.; Mahmoud, K.; Liu, M.K.; Lehtonen, M.; Darwish, M.M. Experimental setup for online fault diagnosis of induction machines via promising IoT and machine learning: Towards industry 4.0 empowerment. *IEEE Access* **2021**, *9*, 115429–115441.
4. Zhang, D. Network Security Middleware Based on USB Key. In Proceedings of the 2008 Fifth IEEE International Symposium on Embedded Computing, Beijing, China, 6–8 October 2008; pp. 77–81. doi:10.1109/SEC.2008.8.
5. Elsisi, M.; Tran, M.Q.; Mahmoud, K.; Mansour, D.E.A.; Lehtonen, M.; Darwish, M.M.F. Towards Secured Online Monitoring for Digitalized GIS Against Cyber-Attacks Based on IoT and Machine Learning. *IEEE Access* **2021**, *9*, 78415–78427. doi:10.1109/ACCESS.2021.3083499.
6. Chu, W. Application of data encryption technology in computer network security. *J. Phys.* **2019**, *1237*, 022049.
7. Li, X. Application of data encryption technology in computer network communication security. *J. Phys.* **2020**, *1574*, 012034.
8. Lee, K.; Yeuk, H.; Choi, Y.; Pho, S.; You, I.; Yim, K. Safe Authentication Protocol for Secure USB Memories. *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl.* **2010**, *1*, 46–55.
9. Tran, M.Q.; Liu, M.K.; Elsisi, M. Effective multi-sensor data fusion for chatter detection in milling process. *ISA Trans.* **2021**, doi:10.1016/j.isatra.2021.07.005.
10. Kaur, M.; Singh, D.; Kumar, V.; Gupta, B.; Abd El-Latif, A.A. Secure and Energy efficient based E-health Care Framework for Green Internet of Things. *IEEE Trans. Green Commun. Netw.* **2021**, *5*, 1223–1231 .
11. Tran, M.Q.; Elsisi, M.; Liu, M.K. Effective feature selection with fuzzy entropy and similarity classifier for chatter vibration diagnosis. *Measurement* **2021**, *184*, 109962.
12. Liao, T.L.; Wan, P.Y.; Chien, P.C.; Liao, Y.C.; Wang, L.K.; Yan, J.J. Design of High-Security USB Flash Drives Based on Chaos Authentication. *Electronics.* **2018**, *7* , 82, doi:10.3390/electronics7060082.
13. Kuamr Nanda, P.; Prasad Das, S.; Ranjan Panda, S. and Singh, D. Impact of Structural Aspect, Metal Gate and Channel Material on UTB-SOI-MOSFET. *Int. J. Innov. Technol. Explor. Eng.* **2019**, *9*, 1638.
14. Pham, D.V.; Syed, A.; Mohammad, A.; Halgamuge, M.N. Threat analysis of portable hack tools from USB storage devices and protection solutions. In Proceedings of the 2010 International Conference on Information and Emerging Technologies, Karachi, Pakistan, 14–16 June 2010 ; pp. 1–5. doi:10.1109/ICIET.2010.5625728.
15. Kaur, M.; Kumar, V. Beta chaotic map based image encryption using genetic algorithm. *Int. J. Bifurc. Chaos* **2018**, *28*, 1850132.
16. Jeong, H.; Choi, Y.; Jeon, W.; Yang, F.; Lee, Y.; Kim, S.; Won, D. Vulnerability analysis of secure USB flash drives. In Proceedings of the 2007 IEEE International Workshop on Memory Technology, Design and Testing, Taipei, Taiwan, 3–5 December 2007; pp. 61–64. doi:10.1109/MTDT.2007.4547620.
17. Zhong, Y.; Yamaki, H.; Yamaguchi, Y.; Takakura, H. Charging Me and I Know Your Secrets! Towards Juice Filming Attacks on Smartphones. In Proceedings of the 1st ACM Workshop on Cyber-Physical System Security, Kyoto, Japan, 22–26 July 2013. doi:10.1109/COMPSAC.2013.6.
18. Meng, W.; Lee, W.; Liu, Z.; Su, C.; Li, Y. Evaluating the Impact of Juice Filming Charging Attack in Practical Environments. In Proceedings of the 20th Annual International Conference on Information Security and Cryptology (ICISC), Seoul, Korea, 29 November–1 December 2017.
19. Noyes, D.; Liu, H.; Fortier, P. Security analysis and improvement of USB technology. In Proceedings of the 2016 IEEE Symposium on Technologies for Homeland Security (HST), Waltham, MA, USA, 10–11 May 2016; pp. 1–3. doi:10.1109/THS.2016.7568955.
20. Nissim, N.; Yahalom, R.; Elovici, Y. USB-based attacks. *Comput. Secur.* **2017**, *70*, 675–688. doi:10.1016/j.cose.2017.08.002.
21. He, F. USB Port and power delivery: An overview of USB port interoperabiliy. In Proceedings of the USB Port and power delivery: An overview of USB port interoperabiliy, Chicago, IL, USA, 18–20 May 2015; pp. 1–5. doi:10.1109/ISPCE.2015.7138710.
22. Sanjaa, B.; Chuluun, E. Malware Detection Using Linear SVM, 2013. Available online: https://www.researchgate.net/publication/261306032_Malware_detection_using_linear_SVM (accessed on 29 November 2021). doi:10.1109/IFOST.2013.6616872.
23. Meng, W.; Lee, W.H.; Murali, S.; Krishnan, S. JuiceCaster: Towards automatic juice filming attacks on smartphones. *J. Netw. Comput. Appl.* **2016**, *68*, 201–212.
24. Li, M.; Liu, J. USB key-based approach for software protection. In Proceedings of the 2009 International Conference on Industrial Mechatronics and Automation, Chengdu, China, 15–16 May 2009; pp. 151–153. doi:10.1109/ICIMA.2009.5156582.
25. Jin, R.; Wang, B. Malware Detection for Mobile Devices Using Software-Defined Networking. In Proceedings of the 2013 Second GENI Research and Educational Experiment Workshop, Salt Lake City, UT, USA, 20–22 March 2013; pp. 81–88. doi:10.1109/GREE.2013.24.
26. Vinod, P.; Laxmi, V.; Gaur, M.; Naval, S.; Faruki, P. MCF: MultiComponent Features for Malware Analysis. In Proceedings of the 2013 27th International Conference on Advanced Information Networking and Applications Workshops, Barcelona, Spain, 25–28 March 2013; pp. 1076–1081. doi:10.1109/WAINA.2013.147.

27. Jiang, D.; Hu, G.; Qi, G.; Mazur, N. A fully convolutional neural network-based regression approach for effective chemical composition analysis using near-infrared spectroscopy in cloud. *J. Artif. Intell. Technol.* **2021**, *1*, 74–82.
28. Basavegowda, H.S.; Dagnew, G. Deep learning approach for microarray cancer data classification. *CAAI Trans. Intell. Technol.* **2020**, *5*, 22–33.
29. Xu, Y.; Qiu, T.T. Human Activity Recognition and Embedded Application Based on Convolutional Neural Network. *J. Artif. Intell. Technol.* **2021**, *1*, 51–60.
30. Ghosh, S.; Shivakumara, P.; Roy, P.; Pal, U.; Lu, T. Graphology based handwritten character analysis for human behaviour identification. *CAAI Trans. Intell. Technol.* **2020**, *5*, 55–65.
31. Hu, G.; Chen, S.H.K.; Mazur, N. Deep Neural Network-based Speaker-Aware Information Logging for Augmentative and Alternative Communication. *J. Artif. Intell. Technol.* **2021**, *1*, 138–143.
32. Gupta, B.; Tiwari, M.; Lamba, S.S. Visibility improvement and mass segmentation of mammogram images using quantile separated histogram equalisation with local contrast enhancement. *CAAI Trans. Intell. Technol.* **2019**, *4*, 73–79.
33. Dhanush, V.; Mahendra, A.R.; Kumudavalli, M.V.; Samanta, D. Application of deep learning technique for automatic data exchange with air-gapped systems and its security concerns. In Proceedings of the 2017 International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 18–19 July 2017; pp. 324–328. doi:10.1109/ICCMC.2017.8282701.
34. Mahboubi, A.; Camtepe, S.; Morarji, H. Reducing USB Attack Surface: A Lightweight Authentication and Delegation Protocol. In Proceedings of the 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Shah Alam, Malaysia, 11–12 July 2018; pp. 1–7. doi:10.1109/ICSCEE.2018.8538400.
35. Fernandes, E.; Crispo, B.; Conti, M. FM 99.9, Radio Virus: Exploiting FM Radio Broadcasts for Malware Deployment, 2013. Available online: http://www.earlence.com/assets/papers/fm99_tifs14.pdf (accessed on 29 November 2021). doi:10.1109/TIFS.2013.2259818.
36. Kang, M.; Saiedian, H. USBWall: A novel security mechanism to protect against maliciously reprogrammed USB devices. *Inf. Secur. J. Glob. Perspect.* **2017**, *26*, 166–185.
37. Luqman, M.; Faridi, A.R. An Overview of Security Issues in Fog Computing. In Proceedings of the 2019 6th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 13–15 March 2019; pp. 1157–1162.
38. Khorsand, Z.; Hamzeh, A. A novel compression-based approach for malware detection using PE header. In Proceedings of the 5th Conference on Information and Knowledge Technology, Shiraz, Iran, 28–30 May 2013. doi:10.1109/IKT.2013.6620051.
39. Dube, T.E.; Raines, R.A.; Grimaila, M.R.; Bauer, K.W.; Rogers, S.K. Malware Target Recognition of Unknown Threats. *IEEE Syst. J.* **2013**, *7*, 467–477. doi:10.1109/JSYST.2012.2221913.
40. Wazid, M.; Katal, A.; Goudar, R.; Singh, D.; Tyagi, A.; Sharma, R.; Bhakuni, P. A framework for detection and prevention of novel keylogger spyware attacks. In Proceedings of the 2013 7th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, India, 4–5 January 2013. doi:10.1109/ISCO.2013.6481194.
41. Mishra, N.; Swagatika, S.; Singh, D. An Intelligent Framework for Analysing Terrorism Actions Using Cloud. In *New Paradigm in Decision Science and Management*; Advances in Intelligent Systems and Computing; Patnaik, S., Ip, A.W.H., Tavana, M., Jain, V., Eds.; Springer: Singapore, 2020; pp. 225–235. doi:10.1007/978-981-13-9330-3_21.
42. Singh, D.; Pati, B.; Panigrahi, C.R.; Swagatika, S. Security Issues in IoT and their Countermeasures in Smart City Applications. In *Advanced Computing and Intelligent Engineering*; Advances in Intelligent Systems and Computing; Pati, B., Panigrahi, C.R., Buyya, R., Li, K.C., Eds.; Springer: Singapore, 2020; pp. 301–313. doi:10.1007/978-981-15-1483-8_26.
43. Sanwal, S.; Singh, K. Juice Jacking—A type of Cyber Attack. *Cybernomics* **2020**, *2*, 25–28.
44. Wlosinski, L.G. Mobile Computing Device Threats, Vulnerabilities and Risk Are Ubiquitous. ISACA 201. Available online: https://www.isaca.org/es-es/resources/isaca-journal/issues/2016/volume-4/mobile-computing-device-threats-vulnerabilities-and-risk-are-ubiquitous (accessed on 29 November 2021)
45. Rath, M.; Swain, J.; Pati, B.; Pattanayak, B.K. Network Security: Attacks and Control in MANET. 2018. Available online: https://www.igi-global.com/chapter/network-security/201602 (accessed on 29 November 2021).
46. Sun, C.; Lu, J.; Liu, Y. Analysis and Prevention of Information Security of USB. In Proceedings of the 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 23–26 September 2021; pp. 25–32.
47. Rath, M.; Pati, B. Security Assertion of IoT Devices Using Cloud of Things Perception. *Int. J. Interdiscip. Telecommun. Netw.* **2019**, *11*, 17–31.
48. Cesare, S.; Xiang, Y.; Zhou, W. Malwise—An Effective and Efficient Classification System for Packed and Polymorphic Malware. *IEEE Trans. Comput.* **2013**, *62*, 1193–1206. doi:10.1109/TC.2012.65.
49. O'Kane, P.; Sezer, S.; McLaughlin, K.; Im, E.G. SVM training phase reduction using dataset feature filtering for malware detection. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 500–509.
50. Baldini, G.; Geneiatakis, D. A performance evaluation on distance measures in KNN for mobile malware detection. In Proceedings of the 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), Paris, France, 23–26 April 2019; pp. 193–198.
51. Fan, Y.; Ye, Y.; Chen, L. Malicious sequential pattern mining for automatic malware detection. *Expert Syst. Appl.* **2016**, *52*, 16–25.
52. Kaur, J.; Singh, D.; Kaur, M. A novel framework for drug synergy prediction using differential evolution based multinomial random forest. *Int. J. Adv. Comput. Sci. Appl.* **2019**, *10*, 601–608.

53. Vignesh, B.S.; Babu, M.R. Classifying the malware application in the Android-based smart phones using ensemble-ANFIS algorithm. *Int. J. Netw. Virtual Organ.* **2018**, *19*, 257–269.
54. Kaur, M.; Singh, D.; Kumar, V. Drug synergy prediction using dynamic mutation based differential evolution. *Curr. Pharm. Des.* **2021**, *27*, 1103–1111.
55. Ganesh, M.; Pednekar, P.; Prabhuswamy, P.; Nair, D.S.; Park, Y.; Jeon, H. CNN-based android malware detection. In Proceedings of the 2017 International Conference on Software Security and Assurance (ICSSA), Altoona, PA, USA, 24–25 July 2017; pp. 60–65.
56. Singh, D.; Kumar, V.; Kaur, M.; Jabarulla, M.Y.; Lee, H.N. Screening of COVID-19 suspected subjects using multi-crossover genetic algorithm based dense convolutional neural network. *IEEE Access* **2021**, *9*, 142566–142580.