Princess Sumaya University for Technology
جامعة الأميرة سمية للتكنولوجيا

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

Jordanian Journal of Computers and Information Technology

March 2023 | VOLUME 09 | NUMBER 01 | ISSN 2415 - 1076 (Online) ISSN 2413 - 9351 (Print)

JJCIT

www.jjcit.org

jjcit@psut.edu.jo

An International Peer-Reviewed Scientific Journal Financed by the Scientific Research and Innovation Support Fund

Jordanian Journal of Computers and Information Technology (JJCIT)

The Jordanian Journal of Computers and Information Technology (JJCIT) is an international journal that publishes original, high-quality and cutting edge research papers on all aspects and technologies in ICT fields.

JJCIT is hosted and published by Princess Sumaya University for Technology (PSUT) and supported by the Scientific Research Support Fund in Jordan. Researchers have the right to read, print, distribute, search, download, copy or link to the full text of articles. JJCIT permits reproduction as long as the source is acknowledged.

AIMS AND SCOPE

The JJCIT aims to publish the most current developments in the form of original articles as well as review articles in all areas of Telecommunications, Computer Engineering and Information Technology and make them available to researchers worldwide. The JJCIT focuses on topics including, but not limited to: Computer Engineering & Communication Networks, Computer Science & Information Systems and Information Technology and Applications.

INDEXING

JJCIT is indexed in:

EDITORIAL BOARD SUPPORT TEAM

LANGUAGE EDITOR

Haydar Al-Momani

EDITORIAL BOARD SECRETARY

Eyad Al-Kouz

JJCIT ADDRESS

WEBSITE: www.jjcit.org
EMAIL: jjcit@psut.edu.jo
ADDRESS: Princess Sumaya University for Technology, Khalil Saket Street, Al-Jubaiha
B.O. BOX: 1438 Amman 11941 Jordan
TELEPHONE: +962-6-5359949
FAX: +962-6-7295534

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

AN EARLY DETECTION MODEL FOR KERBEROASTING ATTACKS AND DATASET LABELING

Remah Younisse[1], Mouhammd Alkasassbeh[1], Mohammad Almseidin[2] and Hamza Abdi[1]

(Received: 25-Aug.-2022, Revised: 28-Oct.-2022, Accepted: 22-Nov.-2022)

ABSTRACT

The wild nature of humans has become civilized and the weapons they use to attack each other are now digitized. Security over the Internet usually takes a defensive shape, aiming to fight against attacks created for malicious reasons. Invaders' actions over the Internet can take patterns by going through specific steps every time they attack. These patterns can be used to predict, mitigate and stop these attacks. This study proposes a method to label datasets related to multi-stage attacks according to attack stages rather than the attack type. These datasets can be used later in machine-learning models to build intelligent defensive models. On the other hand, we propose a method to predict and early kill attacks in an active directory environment, such as kerberoasting attacks. In this study, we have collected data related to a suggested kerberoasting attack scenario in pcap files. Every pcap file contains data related to a particular stage of the attack life-cycle and the extracted information from the pcap files was used to highlight the features and specific activities during every step. The information was used to draw an efficient defensive plan against the attack. Here, we propose a methodology to draw equivalent defensive plans for other similar attacks as the kerberoasting attack covered in this study.

KEYWORDS

Kerberoasting attacks, Attack life-cycle, Dataset labeling, Early detection.

1. INTRODUCTION

TData is the modern fuel to train and calibrate machine-learning (ML) and artificial-intelligence (AI) models. Generating the proper data for ML and AI models built for complicated problems requiring high computational powers is not easy. Labeling this data so that it would give the best results is also not easy.

Building intelligent intrusion-detection systems deploying ML and AI models to detect possible threats over computer networks is no longer a luxury. AI intrusion-detection systems ought to detect possible threats early, so that defense plans can occur early and a more secure network is gained.

Intrusions and attacks over the Internet can take many forms and patterns, each with specific distinguishing techniques and goals. Patterns were found in each attack, making it possible to model and categorize the stages and steps attacks can go through. Many models in this context were proposed, such as the MITRE attack model and the Lockheed Martin model. These models aim to put attack stages in a chain of steps, so that defenders can know at which stage the attack is and build defense policies based on the attack's detected stage. Knowing the attack stage can help kill the attack at this stage or mitigate the attack consequences.

ML and AI models should be built to advance the operation of intrusion-detection systems [1]. These models require a properly labeled dataset; the more significant the dataset, the better the model performance. The literature has a shallow amount of datasets in this context. Therefore, this study proposes a method to create and label datasets related to multi-stage attack life-cycle. We also show how the generated data can be used statistically to detect attacks in an active directory server environment. The detection method is built on the awareness of the attack life-cycle.

An AI intrusion-detection system that is aware of the attack stages, not only the attack occurrence, can be more helpful in providing the proper knowledge which a defender needs to act accordingly. This intelligent intrusion-detection system requires that we collect and label the proper datasets for training. A good defense plan can also be used and built based on the attack stages.

1. R. Younisse, M Alkasassbeh and H. Abdi are with Department of Computer Science, Princess Summaya University for Technology, Emails: r.baniyounisse@psut.edu.jo, m.alkasassbeh@psut.edu.jo and h.abdi@psut.edu.jo
2. M. Almseidin is with Department of Computer Science, Tafila Technical University, Email: alsaudi@ttu.edu.jo

Active directory (AD) is a centralized management system for Windows computers and accounts. Hence, attacking (AD) servers can take many strategies, all aiming to take access to other computers and accounts linked to the server [2]. Kerberos authentication is widely used in the AD environment, where the domain controller processes all authentications using authentication tickets known as TicketGranting Tickets (TGTs) and Service Tickets (STs).

The work presented in [3] has detailed the components, the operation and the attacks in an AD environment. The components of Kerberos were summarized into five components. The five components are: the transport layer, agents, encryption keys, tickets and privileged attribute service. The transport layer component uses port 88 for data exchange between agents *via* the TCP or UDP transport-layer. The agents are either client machines, service servers or Key Distribution Centers (KDCs). The KDC is a central server that works to authenticate users and distribute tickets between them, identifying users against services. Encryption keys and tickets can take many forms, each for a particular purpose during the user's authentication process and privileges granted.

A privileged Attribute Certificate (PAC) is an extension of Kerberos tickets that includes valuable information about the privileges of users. According to [3], the attacks in an AD environment can be classified into nine different forms. Every attack was detailed with mitigation and/or prevention techniques. The attack which we are interested in through this study is the "Kerberoasting" attack. This attack was defined as focusing on capturing a service ticket (TGS) from memory, then decrypting the hash of the offline service credential using a password-cracking tool. Kerberoasting takes advantage of service account delivery during the Kerberos authentication process. Users can request a service ticket from a Domain Controller (DC). The DC is the key distribution center (KDC) in the AD utilizing Kerberos. When clients request service tickets for given services from a DC, they use unique identifiers called Principal Service Names (PSNs). To get the Kerberos authentication, SPNs are required to be registered in AD with at least one service logon account [4].

Kerberos is the default and most widely used authentication service in Windows machines. Despite its strength in securely covering authentication and data communication over a non-secure channel, Kerberos has some weaknesses. It is a stateless protocol as TGS and AS servers do not have a memory of previously granted tickets. Password-guessing attacks can provide valuable information for attackers targeting a system that uses Kerberos for authentication. Another weakness is that all the device clocks trying to get authentication from the AS should be synchronized. Furthermore, finally, if an attacker gets access to the "KRBTGT" account, the attacker can take full access to the domain [3].

The Kerbroasting attack can be made in simple steps. First, the attacker should get a valid SPN name. Then, it requests a service ticket and part of the service ticket is encrypted with the target service account's password hash. Weak passwords can easily be de-hashed by the attacker and hence the account can be cracked.

In this study, we focus on the details of creating a "Kerberoasting" attack; during this step, a labeled dataset for the attack was generated. We also aim to study the attack's life-cycle and label the generated dataset based on the attack stages. A methodology to detect the attack at each stage was proposed and detailed.

A cyber kill chain presents a model for understanding the life-cycle of a cyber-attack and helps improve cybersecurity policies [5]. Lockheed Martin has developed a cyber-kill chain model which describes cyber-attack steps. The kill chain has distinct steps describing each cyberattack life-cycle stage. The kill-chain model helps understand how attacks are performed and helps set plans to deal with attacks, such as ransomware and security breaches. An attack is known to follow an ordered sequence of techniques; using the Lockheed model, these steps are reconnaissance, weaponization, delivery, exploitation, installation, command, control and actions [6]. During the reconnaissance stage, the attacker finds its victim and a proper entrance for the victim's systems. Based on the information collected during the first step, the weaponization step comes as a preparation step for the attack using applicable codes and tools. Delivery is the step when a malicious code or tool is planted in the victim's machine. The exploitation step follows, during which the weapon exploits vulnerabilities in the victim's machine, so that a backdoor channel is activated and access to the victim's machine is guaranteed. This is done during the installation step. The last two steps can be considered a stage when the catastrophe and the attacked machine have fallen.

This model was built so that cyber-attack stages can be mapped to each stage in the model. Defense policies are then built based on at which stage the attack is happening [7]-[8]. It is good to mention the MITRE attack [9]. It is a framework built to collect data and information about different cyber-attacks. It also suggests mitigation and protection techniques for the listed attacks.

The contribution of this study is the presentation of the details of creating a "Kerberoasting" attack. The attack traffic details were collected in pcap files using Wireshark and then processed and analyzed. The pcap files were used with CICFlowmeter to generate a labeled dataset for the attack. What distinguishes the generated dataset is that it is labeled according to the attack's stages rather than the attack's type. The attack life-cycle stages were then thoroughly studied and revised to extract the distinguishing features of each stage, so that the attack could be exposed at any stage. A methodology to detect the attack at each stage was proposed and detailed. The attractive aspect of the followed methodology in this study is that it can be mimicked through other studies with similar life-cycle stages to build efficient defensive plans.

This paper is organized as follows: In Section 2, we present some of the valuable works related to the work presented in this paper. In Section 3, the methodology followed to generate the attack, collect and label the dataset is presented. A method to detect the "Kerberoasting" attack is suggested and discussed in sub-section 3.3. Finally, in Section 4, the conclusion of the work is introduced.

2. RELATED WORK

In literature, many models were proposed to build defensive plans against specific attacks. Cyber-attacks can have different types and targets; hence, a general defensive plan which covers all of them is not practical, but taking them in separate cases would make the task more doable.

For example, the work presented in [10] covers the ransomware threats. The work uses Lockheed Martin's cyber kill-chain model as a road map, then every step during the study covers a particular stage of the ransomware attack life-cycle. Unlike other studies covering detection, prevention and mitigation plans against ransomware attacks in literature, the plan presented in [10] considers that the attack is not a single stage of action, but goes through different actions and stages and accordingly, their plan was built. The work presented here studies "Kerberoasting" attack with the same attitude as well.

Attacks in multimedia service environments and the existing cyber kill-chain models were analyzed in [11]. The study efficiently set proper prevention plans for these attacks and used the Lockheed Martin's model to build a defensive policy against internal and external attacks in multi-media service environments. Attacks in an active-directory environment have recently taken considerable attention due to their wide range of collateral damages in the AD environment; i.e., the works in [12][13][14].

The work presented in [2] collected a dataset from event logs of the domain controller, where clients' computers were recorded and divided into many categories. This division was carried out based on the nature of the event; the dataset was not labelled, as labeling the data for intrusion-detection systems is not an easy task. Hence, an unsupervised learning AI model was built for intrusion detection in an AD environment.

Attacks in an AD environment were also investigated and a machine-learning model for Kerberoasting-attack detection was built. The dataset used in this work is non-synthetic, collected from an AD environment of an entire organization with hundreds of daily users. The data was collected from "event 4679", which is the event generated whenever a key distribution center gets a Kerberos Ticket Granting Service (TGS) request only; so the dataset was not rich in size or features. SVM machine-learning models which used the collected dataset were built later for intrusion detection in the AD environment [4]. Table 1 includes a summary of related literature.

Studying the attack life-cycles has brought forth authoritative results in many cases. The attacks on IoT networks have become a significant concern in cybersecurity in the past few years. The study of [15] has shown that such attacks, for example, take a spiral rather than a sequential form. This information can help build detection and defense strategies against the attack. The study proves that a deep sight inside an attack life-cycle can take the defensive party into a better stronger place.

Attack life-cycle was taken into consideration in [16], but not Kerberos nor an AD attack; yet, the

methodology used in the study was inspiring, since an attack model was built based on the disclosed APT attack cases. The model built in [16] gives an overview of APT attack life-cycle and attack techniques and investigates the distinguishing features of the collected data at each stage. The art of telling an attack life-cycle through data collected in pacp files was mentioned in [17], which is also an inspiration that we used in this study. An enormous amount of data was collected in pcap files, then cleaned and looked for attack signatures in the network traffic that map directly to particular phases of the life-cycle. It was focused on specific parts of the life-cycle when searching for attack signatures.

Labeling the dataset based on the attack life-cycle rather than the attack type is a recent area of interest; it was mentioned in [6], generating two multi-stage attacks; a password-cracking and a DDoS attack; two labeled datasets were also created for the attacks. These datasets are expected to be helpful in building powerful intrusion-detection models.

In cyber forensics, the early knowledge of the attacks can help detect and investigate early. The attack life-cycle model of attacks provides clear directions and orientations about the case and about the next steps to be followed during the investigation [18]. According to [18], pcap files generated to record the packets traffic on a network can be of great benefit during digital forensics.

[19] proposed detection scenarios for monitoring the network for a the potential occurrence of a "Kerberoasting" attack. The methodologies were built based on log data and statistical methods without mentioning the attack life-cycle or the dataset used through the analysis. Not to mention some other useful tools which can digest statistics from pcap files, such as CIC flowmeter. These tools can generate a dataset that machine-learning models can use to train and predict intrusions. [20] also greatly analyzed the content of pcap files to detect attacks using DNS tunneling.

The main contribution of this work is to create a dataset for "Kerberoasting" attacks that is aware of the attack life-cycle. The dataset features' values vary from one stage to another during the attack life-cycle. Hence, training AI models with a dataset labeled according to the stage of the attack can give more detailed results, as it can provide conclusions of at what stage the attack is. Knowing the attack stage would improve the followed defense strategies based on the attack life-cycle. The work also presents the methodology followed to collect and label the dataset; hence, it can be followed to create similar datasets for "Kerberoasting" attacks and other multi-stage attacks to improve the detection models built to detect these attacks. These datasets would enhance the quality of the decisions taken and the defense plans against the attacks.

Table 1. Related work summery.

Ref.	Used AI?	Attack	Contribution	Limitations
[4]	Yes, SVM models	Kerberoasting attack	Kerberoasting attack detection using data collected from Event log 4769.	The technique misses detecting some malicious events on the cost of decreasing false positive alarms.
[2]	Yes, unsupervised leaning was used	APT attacks against AD	The attack activity data recorded in the Event logs was used; a new method based on machine learning for detecting attacks was proposed.	False negative can occur if legitimate administrators use CLI tools regularly.
[16]	No	APT attacks	The attack life-cycle was modeled and explained.	-
[17]	No	Attacks in general were studied.	A method to study the attack life cycle through the use of pcap files was proposed	The dataset used was huge.
[19]	No	Kerberoasting attack	A method to study the attack life cycle of Kerberoasting attack through the use of logged data was proposed.	False positive depends on multiple factors.
[15]	Yes	Botnet attacks	A model representing the spiral life-cycle of the attack was used.	The high complexity of the proposed method.

3. DATA COLLECTION AND ANALYSIS

In this section, we discuss how an attack on an active-directory server was generated. We also show

how the data was collected and labeled. The attack life-cycle is also discussed and detailed according to the Lockheed Martin's model attack life-cycle. We specifically discuss the famous "Kerberoasting" attack.

3.1 Attack Generation

The life-cycle of attacks in an active-directory AD environment was discussed in Mitre attack [9] and detailed in [21], where "Kerberoasting" is listed as a sub-technique of steal or forge Kerberos Tickets technique and falls under credential-access tactics. The attack which we have generated in this study was discussed and detailed in [22]. We used our prior knowledge about the attack to create it and collect the traffic data in pcap files to get a deeper understanding of the attack stages and identifying the features at each stage, which will help in diagnosing the attack at early or even late stages.

The attack stages are listed in Figure 1, along with the used tool being listed inside a cloud shape beside the stage; the attack starts by scanning open ports in the AD server. This step aims to investigate running services on the server and is used in many other attacks, detecting that a port-scanning process on a server can be considered an early alarm for potential attacks. In this step, we detected the Kerberos authentication service on port 88; from an attacker point of view, to perform an attack on Kerberos server, we need to know that port 88 at destination side is active and reachable. In the next step we performed another scanning process focusing on ports 139 and 445; this step aims to investigate publicly shared files and is considered a reconnaissance stage also. However, through this study, the second step gave no valuable conclusions.

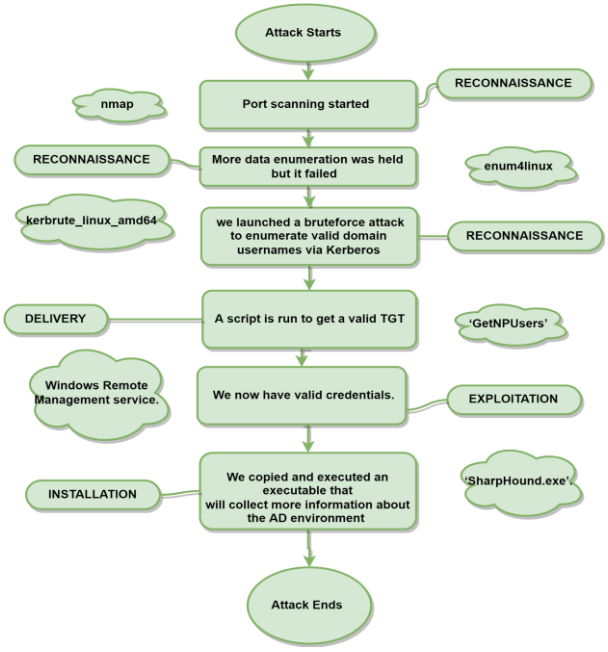


Figure 1. Flowchart of attack stages and tools.

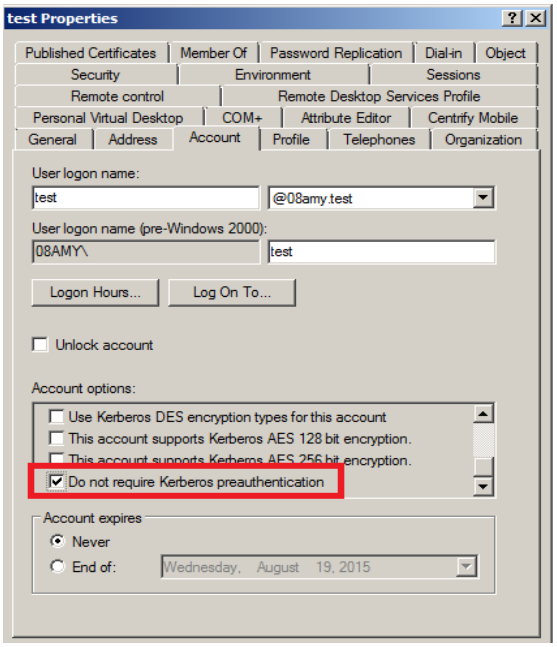


Figure 2. Attack stages and tools.

Next in step 3, knowing that Kerberos is running on the device from, we initiated a brute-force attack on the server using a publicly available dictionary in [23]. The dictionary contains around 100000 famous user names for Windows users; we grabbed a dictionary from the Internet and used it at this step. In step three, the brute-force attack aims to find legitimate user names registered on the Kerberos KD server. One of the valid user names was used in the next step to grant a valid ticket. At step 3 of the attack, we are still in the reconnaissance stage. We target user names whose owners have the property "Do not require Kerberos pre-authentication set (UF_DONT_REQUIRE_PREAUTH)"; this property can be activated by checking the option in the provided test-setting options as shown in Figure 2. These users have configured the Kerberos to consider that whenever they request a "TGT," the requester is the user name owner. Hence, when we requested a password to log into the server, the server responded with a hashed password which we dehashed offline and used to log into this account.

In the last step, we ran an executable file that collected more data about the AD environment. The

software we ran in this step is Sharphound.exe, which a data collector used to collect data from domain controllers and domain-joined Windows systems. The collected data can be information about security-group memberships or domain trusts, abusable rights on active directory objects or group policy links and some other information. The collected information at this step is usually used to facilitate the process of initiating more attacks and take control over the attacked systems.

In Table 2, we list every step with the software tools we used at each step. Also, in Table 2, we map every stage of the attack into a particular stage of the Lockheed Martin's attack stages. We have performed this attack to generate data about the "Kerberoasting" attack, representing an example of a multi-stage attack. We will explain in the following sub-sections how the collected data was used to create a labeled dataset with the label specifying the attack stage. The collected data was also analyzed to investigate for information that can be useful in intrusion detection and prevention strategies.

The attack in the AD environment is divided into stages, each representing a stage in the attack life-cycle model built by Lockheed Martin. Lockheed Martin's model was used in this study due to its simplicity and a direct reflection of multiple stages through the attack life-cycle. The stages and the mapping of Lockheed Martin's model are presented in Table 3.

Table 2. The tools used at each stage of the attack and mapping of Lockheed Martin's model.

Step	Tool	Attack Stage
1	nmap	Reconnaissance
2	enum4linux	Reconnaissance
3	kerbrute	Reconnaissance
4	GetNPUsers	Delivery and then Exploitation
5	winrm and SharpBlood	Installation

In this study, we have not only generated the attack, but also taken care of the attack stages. We knew that every stage of recorded traffic would carry different features and statistics. Hence, the collected data was transformed into CSV files using CIC Flowmeter software while we labeled the data according to the attack stage, as will be detailed in sub-section 3.2. The data exported into the CSV consists of 40 features plus the label column, representing the attack's stage. The listed features represent statistical information about the forward and backward traffic, timestamp and source and destination IP and source ports.

3.2 Data Collection and Labeling

In this sub-section, the methodology followed to collect the dataset is presented. The dataset was created with awareness of the stages the "Kerberoasting" attack will go through. Hence, the related network traffic for each stage was generated in an isolated pcap file using Wireshark. Then, the pcap files were fed into CIC flowmeter to generate the statistics or the features for every stage in a separate CSV file. The data in the CSV files was labeled according to the attack stage. Labeling the dataset according to the attack stage rather than labeling the dataset according to normal or anomaly packets makes the dataset more informative and aware of the stage at which the attack is. The analysis for each stage of the attack can be made with this dataset; it also can be used with AI and ML models which are more precise and specific about the severity of the attack. The presented method is not limited to "Kerberoasting" attack; it can be applied with other attacks to generate different datasets labeled according to the attack life-cycle.

To collect the data, an attack scenario was performed on a server running Kerberos on a virtual machine. In contrast, the traffic on the attacked virtual machine was collected and recorded in pcap files using Wireshark. The information saved in the pcap files was then explored using Wireshark. We also exported the data into CSV files, so that statistical analysis can be carried out on the data to extract a deeper insight into each stage of the attack. CIC flowmeter was used to record statistical information about the packets' flow during the attack stages.

We created the targeted attack from scratch; prior understanding of the attack was beneficial to understand the attack stages and record the network traffic during every stage. We created an actual crime scene, knowing that the crime scene is divided into multiple stages, each with certain attacker behaviors and actions. The attack scenario starts when the attacker uses a tool called nmap to identify

open ports and services. Based on the results returned by nmap, the attacker can notice that there are multiple ports open. The attacker then uses a tool called enum4linux to enumerate more data from the victim's machine. We can tell here that the output was not helpful, but it might be helpful in other scenarios. The attacker also notices that Kerberos was installed on the target's machine as well as other active directory services. The attacker launches a brute-force attack to enumerate valid domain user names *via* Kerberos using the following tool kerbrute_linux_amd64. After identifying a list of valid users, the attacker uses a script called 'GetNPUsers' that exploits a misconfiguration in the AD. This vulnerability is called 'ASREPRoasting'. After receiving a valid TGT, the attacker used a popular wordlist to try and crack the TGT. After successfully cracking the TGT, the attacker now has valid credentials. Using these credentials, the attacker tries to connect to the target machine using the Windows remote management service. After successfully logging in, the attacker copies and executes an executable that will collect more information about the AD environment called 'SharpHound.exe'. The returned file was copied back to the attacker's machine in order to import it into a tool called Bloodhound.

The same mentioned steps can be repeated to generate the attack. The resulting datasets can vary according to the settings on the attacked computer and the Kerberos valid user names. Nevertheless, in case that we guarantee that the Kerberos user name exists inside the used dictionary during the brute-force attack step and Kerberos pre-authentication is disabled on the attacked machine, we can tell that a similar, but not exact dataset will be generated. The attacker's computer can speed up or slow down the attack. The specifications of the network or the status which the attacker is using to access the victim's computer can result in slowing down or speeding up the attack time.

The steps, the tools and the comings of the scenario described here will be detailed and discussed later through the following sub-sections. The network traffic data of every stage of the attack was recorded into a separate pcap file. So, we have a pcap file containing information about the traffic at the reconnaissance, delivery and exploitation stages. Even stages can be divided into sub-stages; for example, the first sub-stage of the reconnaissance stage and the second sub-stage of the reconnaissance stage. This style of recording data about the attack stages can provide information not only about whether the attack is related to an anomaly or a regular packet, but also tell to what stage of the attack the packet can be related when we use the dataset with AI models for example. The literature is rich in labeled datasets for different types of cyber-attacks, while a shallow number of attack datasets are labeled according to the attack life-cycle stages.

The steps followed through this study, which are listed in Figure 1, summarize the attack steps and show that the attack covers four stages; reconnaissance, delivery, exploitation and installation. We also can realize that the reconnaissance step is divided into three steps. During data collection, we created the attack, followed the listed steps and recorded the traffic for each step in a pcap file. We also exported every pcap file into CIC flowmeter and created a labeled dataset for the attack, where the label is a keyword representing the stage to which the traffic belongs.

A labeled dataset containing information about the attack life-cycle can significantly benefit building machine-learning models that can predict the attack and the attack stage, so that prevention and mitigation plans can be drawn according to these stages. The steps we followed to create a dataset for an attack that is aware of the attack stages rather than the attack type can be followed for other attacks. More datasets can be created following the proposed method. This will help generate more rich datasets for intrusion-detection and prevention intelligent models.

During the data collection part of this study, we successfully generated and simulated the "Kerberoasting" attack. The attack was performed and recorded based on the understating of the attack stages. Every stage traffic was saved in a separate pcap file; unlike other datasets, which care about the type of the generated attack, we considered the attack stages the features of which differ for the same attack.

3.3 The Suggested Detection Method

The method used in this study to analyze the collected data was to investigate the packets at every stage in isolation. Finding distinguishing features in every stage and recording these features in the packet's content allow that rapid detection and prevention strategies can be set.

Attacks in the active directory, like most other attacks, start with the reconnaissance stage. As we can see in Table 3, dring the first step of the attack, 133225 packets were sent to the AD server in 3 minutes, which means that the flow was around 37 packets/sec. The fourth column in Table 3 shows the number of distinct destination ports collected during every attack stage. 65510 ports were scanned during the first three minutes of the attack. On average, every port was scanned twice during this stage. The count of the records and the number of destination ports were gathered by counting the tuples from the CSV files created for each stage of the attack. The time taken was concluded from the "timestamp" field listed in the CSV files. In table 3, the notes column is for the researchers' notes which were visually realized, since we have spent a reasonable time studying the distinguishing features and details for the generated CSV files.

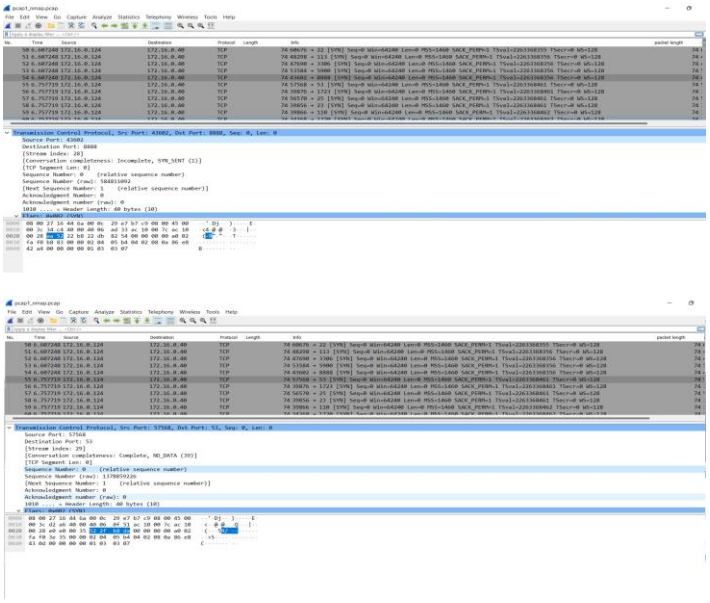


Figure 3. Packet content of two consecutive packets during the port-scanning stage.

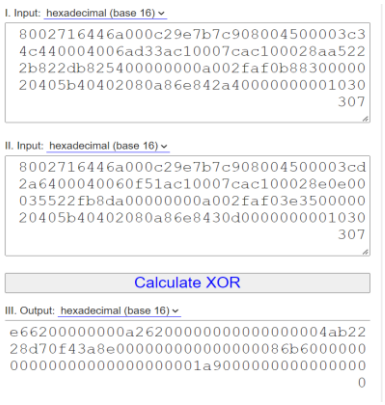


Figure 4. XOR operation between two consecutive packets.

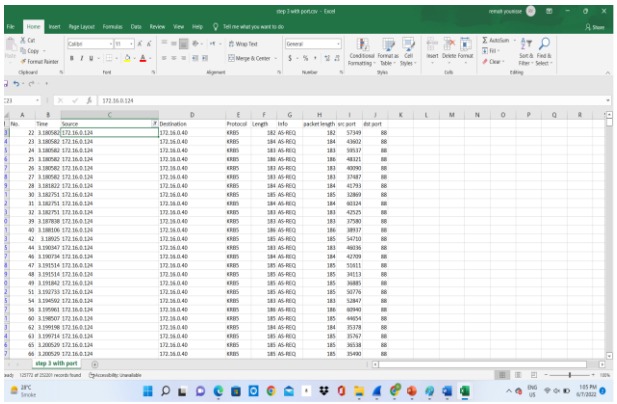


Figure 5. Data collected during brute-force attack on Kerberos.

This behavior of scanning more than 65000 ports in 3 minutes by the same IP address should be a strong indication of the reconnaissance stage of the attack life-cycle. This behavior was detected through our study in two places. First, when we analyzed the pcap files using Wireshark; Figure 3 illustrating the packet content showed high similarities between packets' contents, except for the fields; port numbers, the sequence number of the packet and timestamp value. Figure 4 shows the XORing result between two packets during this attack stage. One suggested method to detect the attack at this stage is performing an XOR operation between packet contents, excluding the fields mentioned earlier. Table 3 shows that the second stage of the attack took only 12 seconds and focused on scanning ports 139 and 445, where the shared data on Windows can be accessed. This step provides shallow information during the intrusion-detection process, but yet can be considered as an indication of scanning stage of a cyber-attack.

Table 3. Attack stages' traffic summery.

Step	Time taken	Num. of Records	Num. of dst. ports	Notes
1	3.0 min	133225	65510	Most packets are of the same length.
2	12.0 sec	555	31	Mostly the scanned ports were 139 and 445.
3	2.0 min	252201	27890	Port 88 was contacted 129534 times.
4	4.0 sec	113	19	-
5	3.0 min	2235	19	-

The third stage shows dense traffic on the network, especially packets connecting to port 88, which is reserved for Kerberos. In Figure 5, it is clear that during the two minutes of running kerbrut, the packets were almost of the same length and the "DST port" value was 88, which is another indication of potential malicious purposes that can be used against the IP address "172.16.0.124". The extracted CSV files' "info" field contains information about the type of the exchanged messages.

No suspicious actions could be detected during stage 4, since the actions performed during this stage are a typical communication process handling a hashed password to the requester. We should re-mention here that in this stage, the attackers could get the password of users who used the "do not require Kerberos pre-authentication" option with their accounts' settings. The KDE sent an encrypted password and the attackers decrypted it offline. Finally, in stage 5, the information field contains the "SharpHound.exe" files which can be considered a keyword for potential attacks. Intrusion-detection systems should be built to consider such keywords to protect network systems.

4. CONCLUSION

In this work, we have presented the methodology to create a "Kerberoasting" attack and collect the traffic data using Wireshark. The collected data was based on the attack stage; so every managed pcap file contains information about a particular attack stage. Then, these pcap files were analyzed and studies were used to generate a labeled dataset for the attack. The labels are a reflection of the attack stage. Every stage of the attack was studied and analyzed, so that the unique characteristics of the attack stage are highlighted. The detection and defense plans against the "Kerberoasting" attack were based on these characteristics. At the early stages of the attack, we realized that the port-scanning process was evident as the destination port varied rapidly in a concise amount of time. Meanwhile, packets of the same size and data have been sent to the destination side with a different port number every time. At the later stage of the attack, a rapid amount of data was sent to port 88, trying to guess a valid user name. Through this work, we detailed how we could extract and use such information at every stage of the attack to build our defenses against the attack. We performed the attack and then studied its data and features to understand the threat in depth. The labeled dataset that we have produced can be used to build intelligent machine-learning models aware of the attack stages and life-cycle. The collected data was also exported into excel sheets and statistical analysis was carried out. In the future, we wish to use the collected data to build the intelligent intrusion-detection model that we have just mentioned.

REFERENCES

[1] A. Yeboah-Ofori et al., "Cyber Threat Intelligence for Improving Cyber Supply Chain Security," Proc. of the IEEE Int. Conf. on Cyber Security and IoT (ICSIoT), pp. 28–33, Accra, Ghana, 2019.

[2] W. Matsuda, M. Fujimoto and T. Mitsunaga, "Detecting APT Attacks against Active Directory Using Machine Leaning," Proc. of the IEEE Conf. on Application, Information and Network Security (AINS), pp. 60–65, Langkawi, Malaysia, 2018.

[3] C. D. Motero et al., "On Attacking Kerberos Authentication Protocol in Windows Active Directory Services: A Practical Survey," IEEE Access, vol. 9, pp. 109289–109319, 2021.

[4] L. Kotlaba, S. Buchovecká and R. Lórencz, "Active Directory Kerberoasting Attack: Detection Using Machine Learning Techniques," Proc. of the 7th Int. Conf. on Information Systems Security and Privacy (ICISSP 2021), pp. 376-383, DOI: 10.5220/0010202803760383, 2020.

[5] M. Alkasassbeh et al., "Detecting Distributed Denial of Service Attacks Using Data Mining Techniques," Int. J. of Adv. Comp. Sci. and Appli., vol. 7, no. 1, 2016.

[6] M. Almseidin, J. Al-Sawwa and M. Alkasassbeh, "Generating a Benchmark Cyber Multi-step Attacks

Dataset for Intrusion Detection," J. of Intelligent & Fuzzy Systems, vol. 43, no. 3, pp. 3679-3694, 2022.

[7] M. Lehto, "APT Cyber-attack Modeling: Building a General Model," Proc. of the 17th Int. Conf. on Cyber Warfare and Security, vol. 17, DOI: 10.34190/iccws.17.1.36, 2022.

[8] M. Almseidin, J. Al-Sawwa and M. Alkasassbeh, "Anomaly-based Intrusion Detection System Using Fuzzy Logic," Proc. of the IEEE Int. Conf. on Inf. Tech. (ICIT), pp. 290–295, Amman, Jordan, 2021.

[9] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington and C. B. Thomas, "Mitre ATTACK: Design and Philosophy," Project No.: 10AOH08A-JC, The MITRE Corporation, 2018.

[10] T. Dargahi, A. Dehghantanha, P. N. Bahrami, M. Conti, G. Bianchi and L. Benedetto, "A Cyber-kill-chain Based Taxonomy of Crypto-ransomware Features," Journal of Computer Virology and Hacking Techniques, vol. 15, no. 4, pp. 277–305, 2019.

[11] H. Kim, H. Kwon and K. K. Kim, "Modified Cyber Kill Chain Model for Multimedia Service Environments," Multimedia Tools and Applications, vol. 78, no. 3, pp. 3153–3170, 2019.

[12] R. Badhwar, "Advanced Active Directory Attacks and Prevention," in Book: The CISO's Next Frontier, pp. 131–144, Springer, 2021.

[13] S. Muthuraj, M. Sethumadhavan, P. Amritha and R. Santhya, "Detection and Prevention of Attacks on Active Directory Using SIEM," Proc. of the Int. Conf. on Information and Communication Technology for Intelligent Systems (ICTIS 2020), vol. 196, pp. 533–541, 2020.

[14] T. Osmëni and M. Ali, "Exploration of the Attacking Web Vectors," Proc. of the IEEE Int. Conf. on Computing, Networking, Telecomm. & Eng. Sci. Appl. (CoNTESA), pp. 31–35, Tirana, Albania, 2021.

[15] A. Hassanzadeh and R. Burkett, "SAMIIT: Spiral Attack Model in IIoT Mapping Security Alerts to Attack Life Cycle Phases," Proc. of the 5th Int. Symposium for ICS & SCADA Cyber Security Research, pp. 11–20, DOI: 10.14236/ewic/ICS2018.2, 2018.

[16] M. Li, W. Huang, Y. Wang, W. Fan and J. Li, "The Study of APT Attack Stage Model," Proc. of the IEEE/ACIS 15th Int. Conf. on Computer and Inf. Sci. (ICIS), pp. 1–5, Okayama, Japan, 2016.

[17] J. D. Mireles, J.-H. Cho and S. Xu, "Extracting Attack Narratives from Traffic Datasets," Proc. of the IEEE Int. Conf. on Cyber Conflict (CyCon US), pp. 1–6, Washington, USA, 2016.

[18] A. Dimitriadis, N. Ivezic, B. Kulvatunyou and I. Mavridis, "D4i-digital Forensics Framework for Reviewing and Investigating Cyber Attacks," Array, vol. 5, p. 100015, 2020.

[19] L. Kotlaba, S. Buchovecká and R. Lórencz, "Active Directory Kerberoasting Attack: Monitoring and Detection Techniques," Proc. of the 6th Int. Conf. on Inf. Sys. Security and Privacy (ICISSP 2020), pp. 432–439, DOI: 10.5220/0008955004320439, 2020.

[20] M. Al-Kasassbeh and T. Khairallah, "Winning Tactics with DNS Tunnelling," Network Security, vol. 2019, no. 12, pp. 12–19, 2019.

[21] MITRE, "Active Directory," [Online], Available: https://attack.mitre.org/datasources/DS0026/, 2022.

[22] MITRE, "Use Alternate Authentication Material," [Online], Available: https://attack.mitre.org/techniques/T1558/, 2022.

[23] SecLists, "Common-credentials," [Online], Available: https://github.com/danielm iessler/SecLists/blob/master/Passwords/Common-Credentials/10-million-password-list-top-100000.txt, Accessed: Oct. 2022.

ملخص البحث:

يتخذ أمان الإنترنت عادةً شكلاً دفاعيّاً للتّصدّي للهجمات ذات الأغراض الخبيثة. ويمكن لأفعال المهاجمين على الإنترنت أن تتّخذ أنماطاً معينة وتأخذ خطواتٍ محدّدةً في كلّ مرّة ينفّذون فيها هجماتهم. ومن الممكن استخدام تلك الأنماط لتوقّع تلك الهجمات والتّصدّي لها وإيقافها. تقترح هذه الدّراسة طريقةً لوسم مجموعات البيانات فيما يتعلّق بالهجمات متعدّدة المراحل، تبعاً لمراحل تلك الهجمات بدلاً من نوع الهجوم. ويمكن استخدام مجموعات البيانات لاحقاً في نماذج تعلُّم الآلة لبناء أنظمةٍ دفاعيةٍ ذكيةٍ. ومن جهة أخرى، نقترح في هذه الدّراسة طريقةً لتوقّع الهجمات وقتلها مبكّراً في بيئة "الدّليل النّشط"، مثل الهجمات الّتي تهدف الى كَسْر كلمة السّرّ للوصول الى البيانات. في هذه الدّراسة، قمنا بجمع البيانات المتعلّقة بسيناريو مقترح لإحدى تلك الهجمات في ملفّات خاصّة، بحيث يحتوي كلٌّ من تلك الملفّات البيانات المتعلّقة بمرحلةٍ معينةٍ من مراحل "دوْرة حَياة" الهجْمة، وتمّ استخدام المعلومات المستخلصة من الملفّات لتسليط الضّوء على السِّمات والانشطة خلال كلّ خطوة. وتمّ استخدام تلك المعلومات في وضع خطّة دفاعية ضدّ الهجمة. كذلك نقترح منهجيةً لوضع خططٍ دفاعية لهجمات أخرى مشابهة غير هجمات كسْر كلمة السّرّ المستهْدَفة في هذه الدّراسة.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

A THREE-BAND PATCH ANTENNA USING A DEFECTED GROUND STRUCTURE OPTIMIZED BY A GENETIC ALGORITHM FOR THE MODERN WIRELESS MOBILE APPLICATIONS

Khadija Abouhssous[1], Layla Wakrim[2], Asmaa Zugari[3] and Alia Zakriti[1]

(Received: 30-Oct.-2022, Revised: 1-Jan.-2023 and 8-Jan.-2023, Accepted: 9-Jan.-2023)

ABSTRACT

This paper presents a design and optimization approach for a tri-band miniature planar rectangular patch antenna structure for wireless mobile applications. The tri-band operation while maintaining a compact size has been achieved by introducing a defected ground structure (DGS) to control the surface current distribution on the patch antenna and consequently achieve multi-band operation. The geometry of the patch and the position of the DGS were optimized by a genetic algorithm to achieve the desired performance using a simple and miniature design with a size of 16 mm × 20 mm × 1.6 mm, an 82% reduction in the size occupied by a conventional single-band structure used in the optimization process. The proposed GA-optimised antenna provided tri-band operation with bandwidths for |S11| > 6 dB from 3.2 - 3.5 GHz, 5.5 - 5.9 GHz and 6.3 - 7.1 GHz. At the centre frequencies of 3.4, 5.7 and 6.7 GHz, the peak gains were 0.7, 1.76 and 2.93 dB, respectively. The optimally designed antenna is etched on an FR-4 substrate. Simulation and measurement results show good agreement, making the proposed structure a suitable candidate for mobile applications requiring small and multifunctional telecommunication devices.

KEYWORDS

Defected ground structure, Genetic algorithm, Tri-band, Miniaturization, Wireless mobile applications.

1. INTRODUCTION

Due to the accelerated development of wireless communication systems, new systems are needed that can operate in multiple frequency bands and accommodate multiple standards. Multiband RF components, which consist of single circuits and operate at several specific frequencies, provide compact solutions for modern wireless communication systems. Due to their advantages, such as lower manufacturing cost, simpler geometry, easier fabrication and integration, microstrip antennas are competitive candidates for many wireless applications. However, these are generally restricted to mono-band operation and their bandwidth is relatively narrow in comparison to other antenna types. Various approaches to multiband antenna design have been developed and studied. For example, a typical technique is to make slots on the radiation patch of the microstrip antenna. As reported in [1], etching two MIM (metal-insulator-metal) rings on a square patch have been shown to change the surface current distributions, effectively improving the radiation and forming a multiband antenna. Other commonly used methods to realize multiband antennas include the addition of parasitic coupling units [2], the insertion of stub-resonant elements [3], introduction of a DGS structure in the form of a squared ring slot incorporating multiple asymmetrical vertical slots [4], the use of two F-shaped resonators with a patch truncated from its central point [5], the use of etched metamaterial with a spiral structure to behave as a complementary split ring resonator (CSSR) antenna [6], as well as the adoption of fractal iteration technology [7]. Among the major disadvantages of all these types of antennas is the size, which is an area of competition for antenna designers, in addition to the narrow bandwidth and the complexity of implementation. Furthermore, to achieve an optimal design and to improve the patch antenna performance, many different optimization algorithm methods are used, such as particle swarm optimization (PSO) [8] and genetic algorithms (GAs) [9]-[10].

1. K. Abouhssous and A. Zakriti are with Lab. of Sci. and Advanced Tech., Department of Civil and Industrial Sci. and Tech., National School of Applied Sci., Abdelmalek Essadi Uni., Morocco. Emails: abouhssoukhadija@gmail.com and azakriti@uae.ac.ma
2. L. Wakrim is with Lab. of Innovation in Management and Engineering for Business (LIMIE), Higher Institute of Engineering and Business (ISGA), Marrakech, Morocco. Email: Layla.wakrim@isga.ma
3. A. Zugari is with Inf. and Telecomm. System Lab., Abdelmalek Essadi Uni., Tetouan, Morocco. Email : azugari@uae.ac.ma

The GA is an evolutionary learning approach similar to the learning of beings. The GA concept is based on the Darwinian evolution theory, according to which organism populations develop by natural selection by transmitting to their descendants variations which allow for survival and replication. However, the potential solutions in GA-based optimization are registered as individuals in the population that need to progress to better solutions. Indeed, an individual's parameters defining a proposed solution are coded like genes on the chromosome. During the optimization process, the GA repeatedly explores the search space and achieves a number of optimized solutions using bio-inspired operations, such as selection, crossover and mutation. At each step, the less adapted individuals of the previous population are substituted by the more adapted ones and the more suitable individuals are selected as the next-generation population. This process is iterated until the final requirement is achieved and therefore the optimal structural parameters are obtained.

This paper presents a conventional patch antenna adapted by using notches, operating at 3.5 GHz.The proposed-structure parameters are calculated by mathematical equations based on the theoretical studies of the patch antennas [11]. The design results show that the overall size of the proposed structure is $40mm \times 45mm \times 1.6mm$, which is a bulky size, with a limited operation in a single frequency band and narrow bandwidth. To overcome these problems and improve the conventional antenna performance in terms of miniaturization, multi-frequency and broadband operation, a defective ground structure (DGS) in the form of a rectangular slot with T-shaped heads on the edges is introduced. The main interest of this work is the application of the genetic algorithm to determine the location of the slot on the ground plane, as well as to optimize the antenna parameters. The results of the optimization by the genetic algorithm show a miniaturization rate of 82% compared to the conventional structure with a tri-band operation and improved bandwidths. A prototype of the optimized tri-band patch antenna has been fabricated and tested in order to validate the procedure. The simulated and measured results show good agreement.

2. METHODOLOGY DESIGN

This section presents an overview of the patch-antenna theory, the Defected Ground Plane-based Structure (DGS) design method and the Genetic Algorithm (GA) based optimization of the antenna parameters and its optimal shape.

2.1 Patch Antenna Theory

The conventional design consists of a rectangular patch printed on a dielectric substrate with a completely metalized bottom side. The substrate has a relative permittivity $\varepsilon_r = 4.4$ and a thickness $h = 1.6\,mm$. The antenna is intended to operate at a frequency of 3.5 GHz.
The transmission-line model implies that for the fundamental mode, along the width of the patch, there is a maximum voltage and a minimum current [11]. This means that the edges along the width of the patch are considered radiating slits. The effect of edges and slot radiation is modeled by an equivalent capacitance and radiation resistance. This extends the dimensions around the periphery of the patch.
The effective patch length and width can be written as [11]:

$$L_e = L + 2\Delta L \tag{1}$$

$$W_e = W + 2\Delta W \tag{2}$$

$\Delta L \; and \; \Delta W$ are the extensions along L and W.

The patch length extension is calculated by the equation:

$$\Delta L = \frac{h}{\sqrt{\varepsilon_e}} \tag{3}$$

The propagation in two media having clearly different permittivities requires the determination of an approximate value of ε_e.

for a band such as $\frac{W}{h} \geq 1$: $$\varepsilon_e = \frac{1}{2}(\varepsilon_r + 1) + \frac{1}{2}(\varepsilon_r - 1)\left(1 + 12\frac{h}{w}\right)^{\frac{-1}{2}} \tag{4}$$

In a rectangular patch antenna, so that an antenna radiates as efficiently as possible, W must be equal to $\frac{\lambda}{2}$ and is calculated by the equation: $$W = \frac{c}{2f_0\sqrt{\frac{(\varepsilon_r+1)}{2}}} \tag{5}$$

where C is the velocity of light in a vacuum.

The effective length is given by the equation: $L_{eff} = \frac{c}{2f_0\sqrt{\varepsilon_{reff}}}$ (6)

where f_0 is the resonance frequency calculated by the equation:

$$f_0 = \frac{c}{2L_e\sqrt{\varepsilon_e}} \quad (7)$$

In this work, the antenna is excited with a notched feed line due to the fact that it is easy to manufacture and simple to adapt to the antenna.

The patch impedance is given by the equation: $Z_a = 90\frac{\varepsilon_r^2}{\varepsilon_r - 1}\left(\frac{L}{W}\right)^2$ (8)

A simple formulation is developed for the calculation of the length of the notch [12]. This value is now constant, since it does not influence the resonance frequency of the antenna.

$l = 10^{-4} \times \frac{L}{2} \times [0.001699\varepsilon_r^7 + 0.13761\varepsilon_r^6 - 6.1783\varepsilon_r^5 + 93.187\varepsilon_r^4 - 682.69\varepsilon_r^3 + 2561.9\varepsilon_r^2 - 4043\varepsilon_r + 6697]$ (9)

This formula is valid for $2 \leq \varepsilon_r \leq 10$.

2.2 Defected Ground Structure (DGS)

The DGS structures are an evolution of the EBG (Electro-Magnetic Bandgap) structures. They are mainly intended for the design of compact and efficient microwave devices. The DGS consists of a defect (etch) in the ground plane of a microstrip transmission line, a coplanar waveguide or any structure where a ground plane exists. Microstrip antennas can be miniaturized by introducing defects in the ground plane [13]. A DGS equivalent circuit is a parallel series tuned circuit with a transmission line that it is coupled to. There are different forms of DGSs that have similar functions and features of the slow wave effect and high-impedance and size miniaturization with the same equivalent circuit [14]. However, to enhance the antenna circuit performance, the DGS structure can be modified or changed. When the DGS is introduced into a microwave antenna, the ground plane etch defect geometry disturbs the current distribution. This disturbance leads to an increase in the effective capacitance and inductance which influence the input impedance and current flow of the antenna [15], thereby reducing its size relative to a resonance frequency and/or causing other resonance frequencies to appear.

2.3 Genetic Algorithm Methodology

The genetic algorithm is a stochastic search technique that is based on Darwin's evolution theory. It is an extremely important approach to solving large search space problems for which conventional approaches are not available and for which the possible solutions lie in a large search space. Gas' most significant benefit over other approaches is their high precision [16]. With the exception of its high computational cost due to GA (which takes time), this technique has the following advantages [17] .

- Possibility to optimize continuous or discrete variables.
- Possibility of operating with a great number of variables.
- Being suitable to be used with numerically generated data, experimental data or analytical functions.

The concepts of biological evolution are used in GAs to solve optimization problems. Gene combination principles in biological reproduction are used to repeatedly modify a population of individual points. Due to its random nature, it increases the chances of identifying a global solution. Consequently, it is extremely efficient and stable in finding globally optimal solutions. A GA's objective is to compute the extrema of an identified function in a data space. An evolutionary process is used to solve a problem using GA, where possible solutions (chromosomes) will be utilized to expand new solutions. Such group of possible solutions will be called a population. For the objective of creating the next generation of the population, only one (particular) population will succeed and be used. The solutions used for creating a new solution (offspring) will be selected based on their fitness function. Which chromosome is most suitable to be reproduced is the best fit.

2.4 Optimization Procedure

The optimization procedure for the proposed design is summarized in the flowchart in Figure 1. We use MATLAB software to implement the genetic algorithm. The advantage of MATLAB is the ability to

invoke external programs; it represents a powerful calculator for complex matrix operations. The VBA script for the antenna design and analysis is available in CST. The genetic algorithm is written in MATLAB by invoking a VBA script in CST that controls the analytical operations on CST Studio. However, the key procedure of the genetic algorithm used to optimize the main parameters of the proposed antenna can be summarized by the steps presented in the flowchart in Figure 2:

Step 1: The identification of the different variables to be optimized and the definition of the maximum and minimum bounds for each variable to be optimized.

Step 2: The optimization process is started by generating a random population, where each individual (in the form of linear vectors) is presented by a chromosome and modeled in the Antenna Toolbox for the calculation of the S_{11} return losses on the ranges of each frequency band. The fitness value of each individual was calculated as shown in (10). Then, these individuals have been sorted by their fitness value.

$$\text{fitness: } Q(x) = \frac{1}{N}\sum_{fmax}^{fmin} Q(f) \tag{10}$$

where $\boldsymbol{S_{11}}$ is the reflection coefficient,

$$Q(f) = \begin{Bmatrix} S_{11} for\ S_{11} \geq -6 \\ -10\ for\ S_{11} < -6 \end{Bmatrix}$$

$$S_{11} = 20log\left|\frac{Z_{in}-Z_C}{Z_{in}+Z_C}\right|$$

$\boldsymbol{Z_{in}}$ is the antenna input impedance and $\boldsymbol{Z_C}$ is the microstrip line characteristic impedance.

Step 3: The choice of the selection procedure and applying it to each variable (each individual of the chromosome), so that the fittest individuals were selected for crossing to reproduce offspring. In addition, additional offspring were obtained by mutations of some random individuals to ensure the exploration of globally optimal solutions. The fitness values of the offspring are calculated.
Step 4: The descendants and previous-generation individuals are selected based on their fitness values. The most suitable individuals are passed onto the next generation. Steps 3 and 4 are repeated until the endpoint is met.

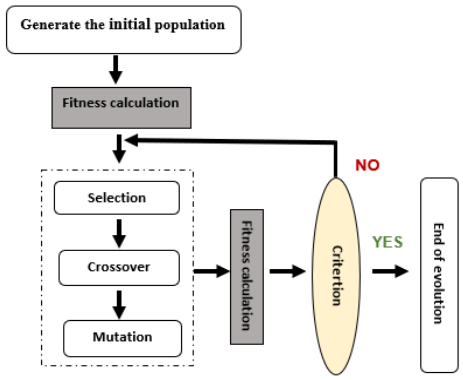


Figure 1. Genetic algorithm organigramme.

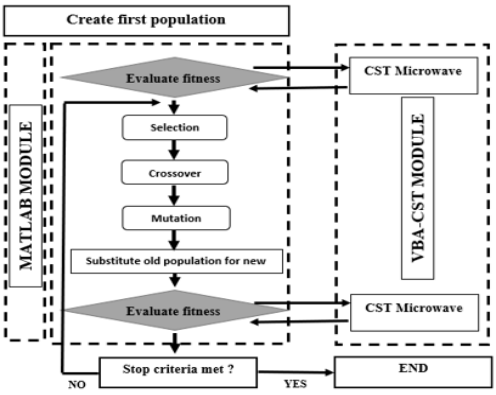


Figure 2. GA's implementation with the CST-MATLAB interface.

2.5 Optimized Tri-band Patch Antenna

The GA parameters, such as population type, population size and the total number of generations, used for the optimization of the geometry parameters of the proposed patch antenna are presented in Table 1 and the optimum parameters resulting from this optimization are presented in Table 2.

A satisfactory solution has been achieved in the 11th-generation optimization, so that the overall size is $16mm \times 20mm \times 1.6mm$, which has a size saving of 82% compared to the conventional structure presented before. The frequency responses of the proposed antenna show a tri-band frequency operation and a bandwidth enhancement with $|S11| > 6\ dB$.

Table 1. Genetic algorithm optimization parameters.

GA Parameter	Value
Number of variables	50
Population type	Double vector
Population size	100
Selection	Roulette
Scaling	Rank
Reproduction elite count	2
Crossover fraction	0.8
Crossover	Single-point crossover
Mutation	Uniform (0.03)
Migration fraction	0.2
Total number of generations	100

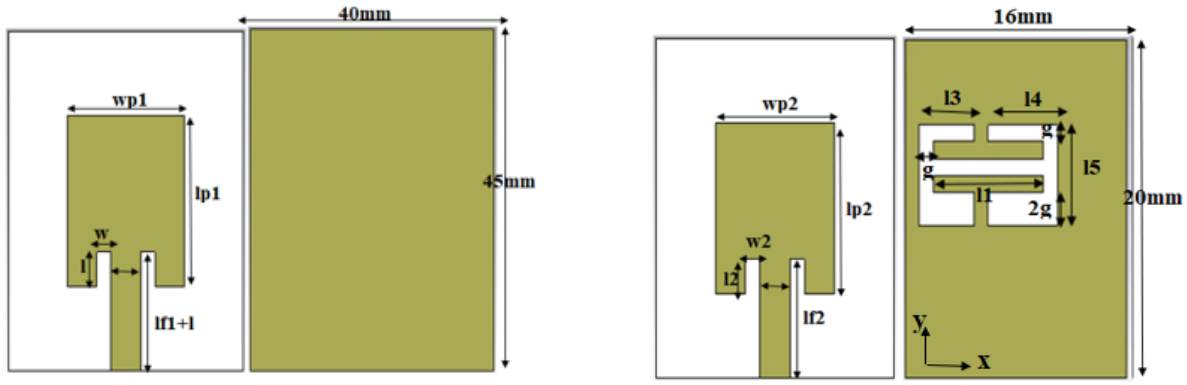
Table 2. Optimized tri-band patch antenna parameters.

Parameter	Value (mm)
w_{p1}	7.179
l_{p1}	10.789
w_{f1}	1.137
l_{f1}	6
w_1	1
l_1	1.8
l_2	8
l_3	6
l_4	5
l_5	4
g	1

3. RESULTS AND DISCUSION

3.1 Antenna Evolution

To describe in detail the design procedure of the proposed patch antenna, two design steps are used, as shown in Figure 3.



Antenna 1 Antenna 2

Figure 3. Antenna evolution steps to achieve the proposed tri-band patch antenna.

Firstly, a rectangular patch antenna is designed [Antenna1]. All design parameters are calculated from the equations in sub-section 2.1. The total size of this antenna is $40mm \times 45mm \times 1.6mm$. The frequency response of this antenna has a single frequency band centred at 3.5 GHz. The return loss $|S11|$ is 27.8 dB, with a bandwidth of 3.46 GHz to 3.53 GHz. The simulation result of the radiation pattern at the resonance frequency shows a maximum gain of 5.37 dB. The performance of this antenna is similar to that of the conventional antenna, with a narrow bandwidth and a bulky size. This antenna serves as a reference design, which has been improved to find the optimal final design.

The next step is to introduce a rectangular slot DGS with T-shaped heads on the edges in the ground plane to provide multi-band functionality and a compact size. All parameters of the patch antenna [Antenna 1] and the position of the DGS on the ground plane are optimized by the genetic algorithm following the process detailed in sub-section 2.4. The results of this optimization give a tri-band and miniature structure [Antenna 2] with a size of $16mm \times 20mm \times 1.6mm$.The return loss | S11| of this optimal antenna is 21 dB, 24 dB and 17.2 dB at 3.4 GHz, 5.7 GHz and 6.7 GHz, with bandwidths of 300 MHz, 400 MHz and 800 MHz, respectively.

A comparison between the simulated characteristics and the performance of the two antennas is presented in Table 3.

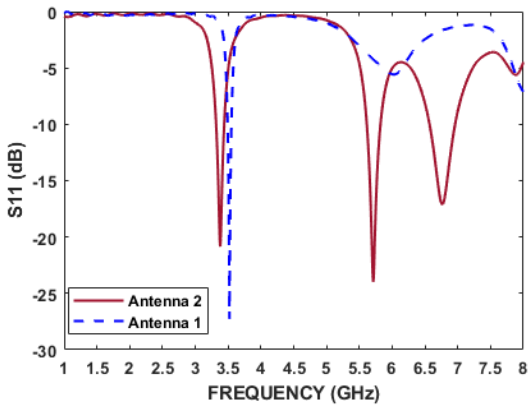


Figure 4. Antenna reflection coefficients.

Table 3. Comparison of antenna characteristics.

	Antenna 1	Antenna 2		
Frequency (GHz)	3.5	3.4	5.7	6.7
S11 (dB)	-27.8	-21	-24	-17.2
BW(MHz)	70	300	400	800
Gain (dB)	5.37	0.7	1.76	2.93
Size (mm^3)	40×45×1.6	16×20×1.6		

3.2 Parametric Study

To confirm the results of the genetic algorithm optimization and the proposed optimal structure of the tri-band patch antenna, a parametric study of the position of the DGS and the depth of the notch is proposed. Each parameter is optimized individually in the optimal solution neighborhoods given by the genetic algorithm; therefore, the same dimensions and the same optimal structure proposed previously are kept and DGS is varied horizontally and vertically on the one hand and the notch depth on the other hand, in such a way that each time two parameters are fixed at the optimal value given by the GA and the third is varied independently. The parameters and optimization conditions of this study are given in Table 4 and some simulated results are presented in Figures 5,6 and 7.

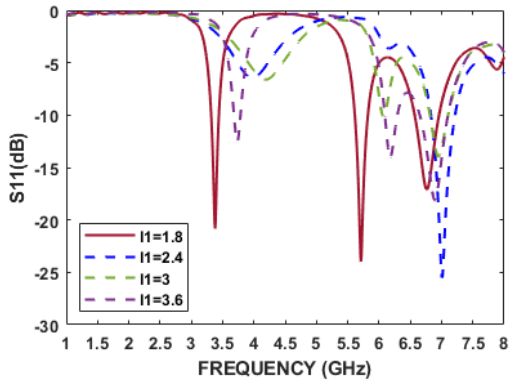


Figure 5. Result of the notch optimization.

Table 4. Optimization parameters.

Parameter		Notch depth	Horizontal position	Vertical position
		L1	x	y
Bounds	Min.	0	-2	-6
	Max.	3.6	2	6
Pat value		0.6	0.5	0.5
Best value		1.8	2	2

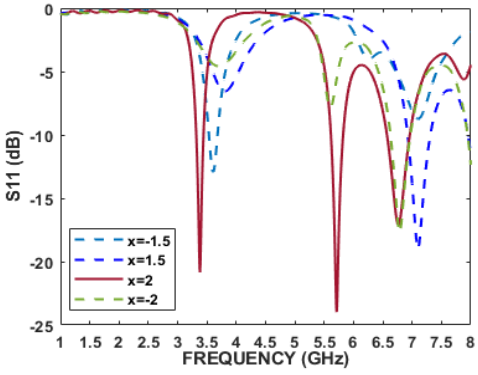


Figure 6. Result of the DGS horizontal position optimization.

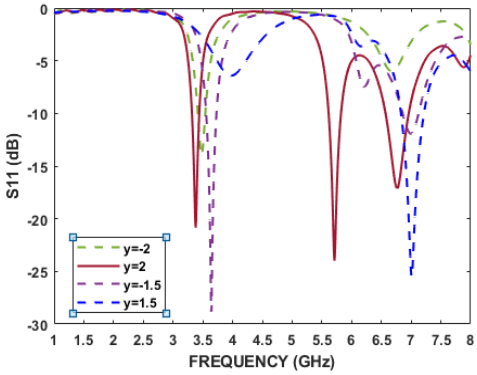


Figure 7. Result of the DGS vertical position optimization.

The first observation on the results of these optimizations is that each variation of these three parameters leads directly to a significant change in the resonance frequencies and multi-band operation, on the adaptation and on the bandwidth. Therefore, for a multidimensional search space composed of several parameters, it is necessary to have a large number of individuals to allow convergence to the optimal or

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

near-optimal solution, which becomes difficult and restricted for traditional optimization. In contrast, the genetic algorithm parameters and results have proved their possibilities and efficiency to converge to the optimal solution and the design of the antenna in an optimal and original or non-intuitive way.

3.3 VSWR: Voltage Standing Wave Ratio

To evaluate the degree of matching or mismatching of the antenna, its Voltage Standing Wave Ratio (VSWR) must be simulated for the required frequency ranges. Figure 8 shows the simulated VSWR for the three resonance frequencies of the optimized antenna. The VSWR has values of 1.2 at 3.4 GHz, 1.25 at 5.7 GHz and 1.8 at 6.7 GHz. These VSWR values are less than 2, which is an acceptable performance according to the simulation results.

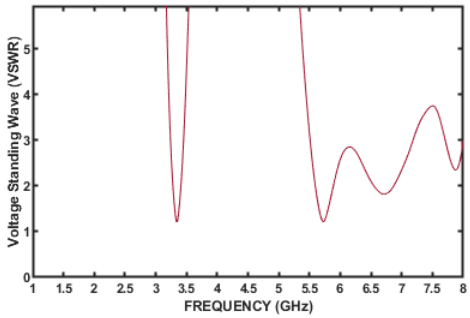


Figure 8. The VSWR of the proposed antenna.

3.4 Radiation Pattern

Figure 9 shows the two-dimensional radiation patterns at the three resonance frequencies. We can see that the proposed patch antenna provides an omnidirectional radiation pattern at 3.4 GHz (Figure 9 (a)) with a peak gain of 0.7 dB and a nearly directional radiation pattern with peak gains of 1.76 dB and 2.93 at 5.7 GHz and 6.7 GHz, respectively (Figure 9 (b) and Figure 9 (c)).

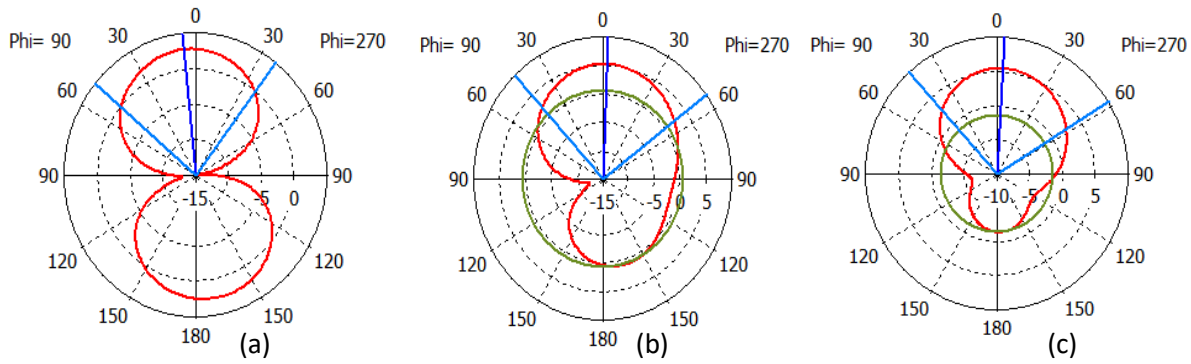
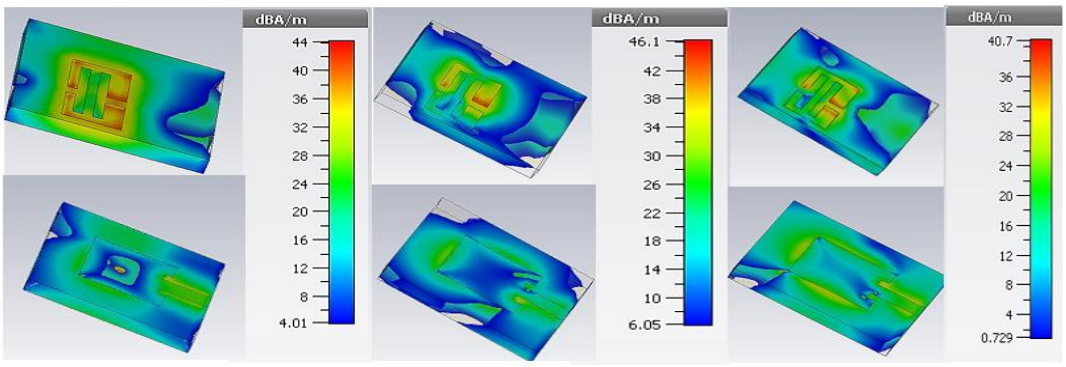


Figure 9. 2D radiation pattern: (a) at 3.4 GHz ;(b) at 5.7 GHz; (c) at 6.7 GHz.



(a) (b) (c)

Figure 10. Simulated surface current distribution of the proposed antenna: (a) at 3.4 GHz; (b) at 5.7 GHz; (c) at 6.7GHz.

3.5 Current Distribution

Figure 10 shows the current distribution for the optimized patch antenna. It is apparent that the current is more concentrated along the DGS ground plane, the thing that explains that defects in the metal ground plane structure disrupt the current distribution, resulting in controlled excitation and propagation of the electromagnetic wave through the substrate layer and a change in the resonance peak.

3.6 Fabrication and Measurement

The structure of the designed antenna is etched on an FR-4 substrate, as shown in Figure 11.

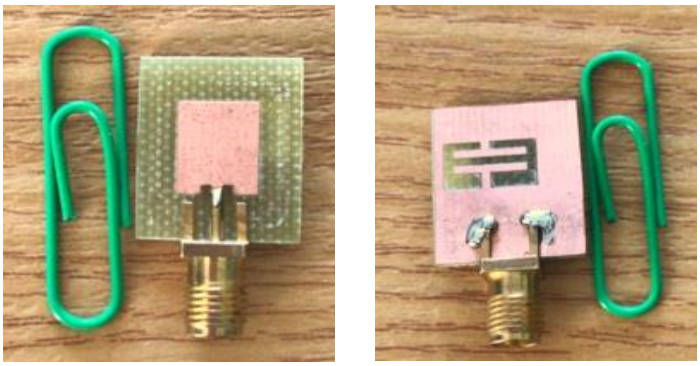


Figure 11. Prototype of optimized tri-band microstrip patch antenna.

The physical design of the fabricated microstrip antenna has a compact size with a high miniaturization rate compared to the conventional antenna. The measurements and frequency responses are shown in Figure 12. It can be seen that the practical results correspond well to the simulated results. Nevertheless, some deviations have been noted, caused by the inaccuracy of the manufacturing and measurement conditions.

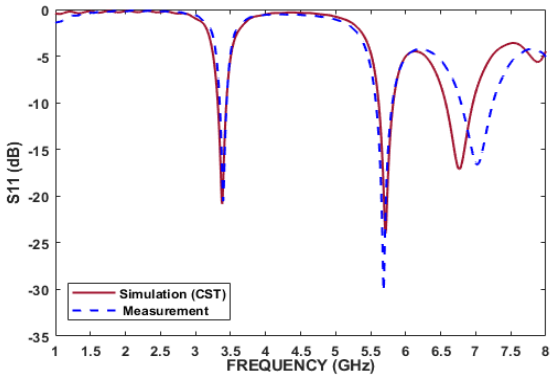


Figure 12. The simulated and measured reflection coefficient of the proposed tri-band patch antenna.

3.7 Comparative Study

To properly evaluate the results of our work, a quantitative comparison between the proposed tri-band antenna and the antennas reported in previously published works in terms of different parameters (resonance frequency, bandwidth, return loss, gain and size) is made. The results of this comparison are given in Table 5. All these designs have been presented in many different forms and techniques to achieve the required characteristics and to cover multiband operation. However, there are many limitations, such as complex structures and large size of designs. Interestingly, in terms of miniaturization and bandwidth, the patch antenna proposed in this study has improved performance over the reference antennas. Furthermore, the performance of the proposed antenna is comparable to those of the reference antennas in terms of return loss and gain. Moreover, the simplicity and ease of fabrication of the structure make it a suitable candidate for mobile applications requiring miniature and multi-band structures that can be easily integrated into modern telecommunication systems.

Table 5. Comparison between the proposed patch antenna and antennas from previous works.

Ref	Freq. (GHz)	$S_{11}(dB)$	BW (MHz)	Gain (dB)	Technology	Size (mm^3)
[4]	2.4	-27	197	2.017	DGS planar patch antenna	$34 \times 30 \times 1.6$
	3.5	-17	118	2.994		
	5.8	-15	90	3.362		
[5]	1.8	-16	140.2	2.34	F-shaped planar patch antenna	$60 \times 50 \times 1.6$
	3.5	-17	180.1	5.2		
	5.4	-19	200.2	1.42		
[6]	1.9	-24.56	130	4.1	Spiral-shaped planar patch antenna	$50 \times 56 \times 1.6$
	2.45	-27.21	290	4.25		
	3.19	-22.46	160	4.74		
This work	**3.4/3.45**	**-21/-19.9**	**300/190**	**0.7**	DGS planar patch antenna	$\mathbf{16 \times 20 \times 1.6}$
	5.7/5.6	**-24/-30.2**	**400/450**	**1.76**		
	6.7/7	**-17.2/-16.5**	**800/900**	**2.93**		

4. CONCLUSION

A tri-band miniature planar rectangular patch antenna structure for wireless mobile applications has been proposed. The miniaturization, tri-band functionality and high performance of the proposed design are achieved by optimizing the structure parameters and the position of a rectangular slot with T-shaped heads on the edges in the ground plane by a genetic algorithm (GA). The antenna performance simulations were performed using an interface between the Antenna Toolbox in MATLAB 2020a and CST studio suite 2019.The proposed tri-band patch antenna's frequency response is well matched to three frequency bands, such that the reflection coefficients are equal to -21dB, -24dB and -17.2dB for the resonance frequencies centered at 3.4 GHz, 5.7 GHz and 6.7 GHz, respectively, with improved bandwidths, a typical radiation pattern and a significant gain. A prototype of the studied antenna is manufactured to validate the optimized structure. The simulated and measured results are almost identical and demonstrate the good performance and tri-band functionality while keeping the simple and compact geometry.

REFERENCES

[1] M. Zavvari, R. Ebadzadeh and M. Mohammadifar, "Localized Surface Plasmons of a Corrugated Metal–insulator–metal Ring Resonator for Enhanced Multiband Antenna," Electronics Letters, vol. 54, no. 3, pp. 120–122, DOI: 10.1049/el.2017.2456, 2018.

[2] T. Dabas, B. K. Kanaujia, D. Gangwar, A. K. Gautam and K. Rambabu, "Design of Multiband Multipolarized Single Feed Patch Antenna," IET Microwaves, Antennas & Propagation, vol. 12, no. 15, pp. 2372–2378, DOI: 10.1049/iet-map.2018.5401, 2018.

[3] Y. F. Cao, S. W. Cheung and T. I. Yuk, "A Multiband Slot Antenna for GPS/WiMAX/WLAN Systems," IEEE Transactions on Antennas and Propagation, vol. 63, no. 3, pp. 952–958, DOI: 10.1109/TAP,2389219, 2015.

[4] A. Ibrahim, N. Arina Fazil and R. Dewan, "Triple-band Antenna with Defected Ground Structure (DGS) for WLAN/WiMAX Applications," J. Phys.: Conf. Ser., vol. 1432, no. 1, p. 012071, DOI: 10.1088/1742-6596/1432/1/012071, Jan. 2020.

[5] A. S. Elkorany et al., "Implementation of a Miniaturized Planar Tri-band Microstrip Patch Antenna for Wireless Sensors in Mobile Applications," Sensors, vol. 22, no. 2, Art. no. 2, DOI: 10.3390/s22020667, Jan. 2022.

[6] N. R. Kumar, P. D. Sathya, S. K. A. Rahim, M. Z. M. Nor, A. Alomainy and A. A. Eteng, "Compact Tri-band Microstrip Patch Antenna Using Complementary Split Ring Resonator Structure," The Applied Computational Electromagnetics Society Journal (ACES), vol. 36, no. 3, pp. 346–353, Mar. 2021.

[7] A. Abed, J. Mandeep and M. Islam, "Compact-size Fractal Antenna with Stable Radiation Properties for Wi-Fi and WiMAX Communications," KSII Transactions on Internet and Information Systems, vol. 12, pp. 2743–2747, DOI: 10.3837/tiis.2018.06.016, Jun. 2018.

[8] K.-C. Lee and J.-Y. Jhang, "Application of Particle Swarm Algorithm to the Optimization of Unequally

Spaced Antenna Arrays," Journal of Electromagnetic Waves and Applications, vol. 20, no. 14, pp. 2001–2012, DOI: 10.1163/156939306779322747, Jan. 2006.

[9] N. Herscovici, M. F. Osorio and C. Peixeiro, "Miniaturization of Rectangular Microstrip Patches Using Genetic Algorithms," IEEE Antennas and Wireless Propagation Letters, vol. 1, pp. 94–97, DOI: 10.1109/LAWP.2002.805128, 2002.

[10] P. Soontornpipit, C. M. Furse and Y. C. Chung, "Miniaturized Biocompatible Microstrip Antenna Using Genetic Algorithm," IEEE Transactions on Antennas and Propagation, vol. 53, no. 6, pp. 1939–1945, DOI: 10.1109/TAP.2005.848461, Jun. 2005.

[11] Constantine A. Balanis, Antenna Theory: Analysis and Design, 2nd Edition, ISBN: 0000471592684, [Online], Available: https://www.amazon.com/Antenna-Theory-Analysis-Design-2nd/dp/0471592684.

[12] M. Ramesh and K. B. Yip, "Design Formula for Inset Fed Microstrip Patch Antenna," Microwaves and RF, vol. 3, pp. 5–10, Dec. 2003.

[13] M. Kumar and V. Nath, "Analysis of Low Mutual Coupling Compact Multi-band Microstrip Patch Antenna and Its Array Using Defected Ground Structure," J. Engineering Science and Technology, vol. 19, no. 2, pp. 866–874, DOI: 10.1016/j.jestch.2015.12.003, Jun. 2016.

[14] Ashwini K. Arya, M. V. Kartikeyan and A. Patnaik, "Defected Ground Structure in the Perspective of Microstrip Antennas: A Review," Frequenz, vol. 64, no. 5–6, pp. 79–84, DOI: 10.1515/FREQ.2010.64.5-6.79, Jun. 2010.

[15] A. Arya, A. Patnaik and K. Machavaram, "Microstrip Patch Antenna with Skew-F Shaped DGS for Dual Band Operation," Progress in Electromagnetics Research M, vol. 19, DOI: 10.2528/PIERM11052305, Jan. 2011.

[16] I. Bouchachi, K. Hamdi-Cherif, K. Ferroudji et al., "A Comparison of Genetic Algorithm and Practical Swarm Optimization for the Design of Waveguide Filters," Proc. of the Int. Telecommunications Conf., vol. 504, pp. 89-98, [Online], Available: https://link.springer.com/chapter/10.1007/978-981-13-0408-8_8, 2022.

[17] R. L. Haupt and S. E. Haupt, Practical Genetic Algorithms, 2nd Ed. Hoboken, N. J.: John Wiley, 2004.

ملخص البحث:

تقدّم هذه الورقة طريقةً لتصميم هوائي صغير مسطّح مستطيل ثلاثي النّطاقات للتّطبيقات المتنقّلة اللاسلكية. وقد تم تحقيق التّشغيل ثلاثي النّطاقات مع الحفاظ على حجم صغير للهوائي عبر تشويه بنية الأساس من أجل التّحكم بتوزيع تيار السّطح. وقد تمّ تحديد الأبعاد الهندسية للرّقعة بالإضافة الى تحديد الموضع المثالي لتشويه بنية الأساس بواسطة خوارزميّة جينيّة بهدف الحصول على أداء مثالي للهوائي باستخدام تصميم بسيط وصغير بحجم بلغ (16×20×1.6) ملم^3؛ أي بنسبة أقلّ من حجم الهوائي أحادي النّطاق مقدارها 82%.

لقد حقّق الهوائي المقترح تشغيلاً ثلاثي النّطاقات (3.2 - 3.5) جيجا هيرتز؛ (5.5 - 5.9) جيجا هيرتز؛ (6.3 - 7.1) جيجا هيرتز عند $|S11| > 6$ ديسيبل. وبلغت قيم كَسْب الهوائي عند الترددات المركزية (3.4 و 5.7 و 6.7) جيجا هيرتز (0.7 و 1.76 و 2.93) ديسيبل على الترتيب. وقد تمّ وضع الهوائي المصمّم بصورة مثالية على طبقة أساس (FR-4). وبينت نتائج المحاكاة ونتائج القياسات اتّفاقاً جيداً، وهذا يجعل البِنية المقترحة مرشحةً للتّطبيقات اللاسلكية المتنقّلة التي تتطلب أجهزة اتّصالات صغيرة ومتعددة الوظائف.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

EFFECTIVENESS OF ZERO-SHOT MODELS IN AUTOMATIC ARABIC POEM GENERATION

Mohamed El Ghaly Beheitt and Moez Ben HajHmida*

(Received: 25-Oct.-2022, Revised: 9-Jan.-2023, Accepted: 13-Jan.-2023)

ABSTRACT

Text generation is one of the most challenging applications in artificial intelligence and natural-language processing. In recent years, text generation has gained much attention thanks to the advances in deep-learning and language-modeling approaches. However, writing poetry is a challenging activity for humans that necessitates creativity and a high level of linguistic ability. Therefore, automatic poem generation is an important research issue that has attracted the interest of the Natural Language Processing (NLP) community. Several researchers have examined automatic poem generation using deep-learning approaches, but little has focused on Arabic poetry. In this work, we exhibit how we utilize various GPT-2 and GPT-3 models to automatically generate Arabic poems. BLEU scores and human evaluation are used to evaluate the results of four GPT-based models. Both BLEU scores and human evaluations indicate that fine-tuned GPT-2 outperforms GPT-3 and fine-tuned GPT-3 models, with GPT-3 model having the lowest value in terms of poeticness. To the best of the authors' knowledge, this work is the first in literature that employs and fine-tunes GPT-3 to generate Arabic poems.

KEYWORDS

Natural-language processing (NLP), Natural-language generation (NLG), Deep learning, Transformer, GPT-2, GPT-3, Arabic poems.

1. INTRODUCTION

Natural-language Generation (NLG) is a challenging topic that has piqued the interest of the Natural Language Processing (NLP) community [1]. Poem generation is an example of NLG that is particularly interesting due to its unique characteristics. One of the most challenging tasks in NLG is automatic poem generation, since poetry is an art form. Nowadays, many researchers are motivated to contribute to automatic poem generation [2][3][4][5][6][7][8][9][10]. Stochastic search [11] and statistical machine-translation models [12] are traditionally recommended for this task. Lately, deep neural networks have been utilized to generate natural-sounding poetry [13]-[14]. Although these models appear to be promising, they are severely limited in various ways. Previous research, for example, frequently fails to retain topic coherence [15]-[16] and increase word variety [14], both of which are essential qualities of poems. Compared to the efforts made in English and Chinese for NLG, Arabic deep-learning applications for NLG, especially Arabic-poetry generation, remain limited.

Arabic poetry is the oldest type of Arabic literature. Poetry has represented the most profound sense of Arab self-identity and collective past and future ambitions in the Arabic literary tradition. Arabic poetry is frequently classified into classical and modern poetry (also named free poetry) [17]. Appropriately, any poetry written in the classical form is referred to as "traditional poetry "or" vertical poetry" as long as it follows the traditional form and structure and the vertical parallel composition of its two components known as hemistichs. A hemistich is a half-line of verse, followed by a second hemistich, making together a verse unit. On the other hand, modern poetry differs from traditional poetry in terms of form, structure, rhyme and subject matter. Arabic poetry is defined as rhymed, metered speech. Table 1 exposes examples of verses from Arabic poetry.

Large language models, like GPT-3 [18], are tens of gigabytes in size, trained on terabytes of text data and have mostly been designed for text generation. GPT-3 was introduced by OpenAI as the largest language model when compared to state-of-the-art language models, with 175 billion trainable parameters [18]. Apart from raw-text generation, GPT-3 can also generate poems, codes, stories, …etc. Brown et al. [18] demonstrated that GPT-3 excelled in various NLP tasks with few-shot learning and even without any fine-tuning.

M. El G. Beheit and *M. Ben HajHmida (Corresponding Author) are with University of Tunis El Manar, Tunis, Tunisia. Emails: medghaly.beheitt@fst.utm.tn and moez.benhajhmida@enit.utm.tn

GPT-2 was succeeded by GPT-3, which utilizes the same transformer architecture. A major distinction between the two models is their size; GPT-3 has 175 billion parameters, significantly more than GPT-2 with 1.5 billion parameters. We are challenged to measure the effectiveness of fine-tuning GPT-2 compared to the use of a zero-shot or few- shot GPT-3 model in the case of Arabic-poem generation. In this work, we compare our previously proposed fine-tuned GPT-2 model [19] to state-of-the-art models as well as some variants of GPT-3 model on Arabic-poem generation. In this comparison, we use automatic evaluation; namely, BLEU scores and human evaluation. To the best of our knowledge, this is the first work in literature that employs GPT-3 to generate Arabic poems.

Table 1. Example of verses from Arabic poems from Arab poet Abu al-Tayyib Ahmad ibn Al-Husayn Al-Mutanabbi [17].

Arabic verses	**English translation**
أنا الذي نظر الأعمى إلى أدبي وأسمعت كلماتي من به صمم	I am who made the blind see my art and made the deaf hear my words
وإذا كانت النفوسُ كباراً تعبت في مرادِها الأجسامُ	If the souls were great The bodies would be tired in achieving their will

We structure the paper as follows. Section 2 describes related state-of-the-art approaches. Section 3 introduces the architectures of models used in this study and Section 4 presents the proposed models. Section 5 illustrates the evaluation methods and we detail experimental results and provide a useful discussion on poem-generation capabilities in Section 6. Finally, Section 7 shows conclusions and future work.

2. RELATED WORK

Poetry composition is undoubtedly the most difficult of the text-generation sub-tasks, because poetry must be written elegantly and ideally according to a particular context. Over the last few decades, automated poem generation has been a popular research area. However, most of the work on poem generation is accomplished in Chinese and English. This section covers recent approaches for generating poems in Chinese and English, as well as a few works in languages similar to Arabic, such as Persian, and finally cites a few existing works in the Arabic language.

Yi et al. developed a model to generate coherent Chinese poetry with a flexible clear description of the poem's topic [20]. They built an encoder-decoder framework based on a bi-directional recurrent neural network (Bi-RNN) with an attention mechanism. They also tested the model on three styles of poetry. The quatrain style is known to be the most difficult form, which is a pair of matching couplets, each line consisting of five or seven syllables. In this work, authors used a corpus of 71,000 quatrains to train a model, while 1,000 quatrains were used for the test. The Working Memory Model was utilized to write a poetry line while keeping the previous line in mind. The previous line is saved in local memory and will be concatenated with the following line. The results of this model were compared to those of state-of-the-art models by poetry experts. The model obtained higher scores on Coherence (3.57) and Relevance (3.77), indicating that it produced poems of higher quality and cohesiveness.

Liu et al. [21] presented a rhetorically controlled encoder-decoder to develop modern Chinese poetry. This model employs a continuous latent variable as a rhetoric controller in an encoder to record distinct rhetorical patterns. Then, it integrates rhetoric-based mixtures while generating modern Chinese poetry. In this model, word embedding, rhetoric label embedding and hidden state sizes are set at 128, 128 and 128, respectively. The latent variable has 256 dimensions and a single-layer decoder is employed. The human evaluation results show that this method achieves the best results in terms of the Meaningfulness (3.2) and Rhetorical Aesthetics (3.5) metrics. Also, experiments reveal that this model can generate Chinese poetry with convincing metaphors and personification.

Deng et al. [22] presented a novel iterative polishing framework for highly competent Chinese poem generation in this research. An encoder-decoder structure is used to generate a poem draft in the first stage. Following that, the authors suggested Quality-Aware Masked Language Model (QA-MLM) to be used to polish the document in terms of linguistics and literalness. QA-MLM can use a multi-task learning system to identify whether polishing is required based on the poetry draft. In this approach, they used corpus for training and testing the models consisting of approximately 130,525 poems with a total of 905,790 lines. BERT was selected as the encoder with 12 layers and initialized with the parameters pre-trained by [23] and the 2-layer transformer decoder was selected for poem generation.

Human and automatic evaluations were performed and the findings show that this approach effectively improves the performance of encoder-decoder structures.

In [24], the authors proposed a generate-retrieve-then-refine paradigm for poetry generation based on the creative pro- cess of humans. It allows a generative model to benefit from generated draft and retrieval outcomes. To increase coherence, the authors employ bidirectional sentence-level context from previously generated lines and draft lines. In addition, they present the "refining vector," which is distilled by the fantastic word recognition process to develop newer and more unique expressions. The authors collected a 263,669 modern Chinese poetry dataset with 9,209,186 sentences. In this approach, they employ an encoder-decoder model with word2vec embeddings. The word embedding size is 128 and the recurrent hidden layers of the encoder and decoder have 128 hidden units and 4 layers. They used the Adam algorithm [25] to train the model, with a batch size of 512 and a learning rate of 3e-4. The results of experiments on Coherence (3.98), Impressiveness (3.86) and Poeticness (3.40) reveal that this model surpasses baselines in terms of consistency and novelty.

Lau et al. [26] built a model based on a variant of an LSTM encoder-decoder with attention [27] for composing English quatrains (Shakespeare-like sonnets). The authors developed a joint design of three neural networks that capture language, rhyme and meter to construct quatrains. They used 3,355 sonnets to train and test these models. They also assessed the quality of generating quatrains using crowdsourcing and expert assessment. According to crowdsourcing and expert evaluations, the poems generated matched the sonnet structure, but lacked readability and coherence.

Santillan and Azcarraga [28] described a method for generating English poems from a given input poem seed utilizing transformers [29] and doc2vec embeddings [30]. This technique uses a pre-trained model, which is then fine-tuned using a poem dataset and assesses generated poem output using a cosine similarity score from a doc2vec model. The corpora used in this approach are divided into two datasets, which are 50-200 characters long (short poem set) and those longer than 200 characters (long poem set), each comprising 58,955 poems, respectively. Moreover, all transformer training employed the same hyper-parameters, such as a learning rate of $2e^{-5}$ and a batch size of 2. This approach ensures strong cohesiveness between the output and the given input text according to the results.

Van de Cruys introduced an automatic poetry-generation system trained just on standard, non-poetic text [10]. The system employs a recurrent neural encoder-decoder architecture that incorporates poetic and topical constraints by changing the neural network's output probability distribution to create potential verses. Then, the best verse is chosen for inclusion in the poem using a global optimization framework. The authors used an encoder and decoder model consisting of two GRU layers with a hidden state of size 2048. The model parameters are optimized using stochastic gradient descent with an initial learning rate of 0.2 and a batch size of 64. They also trained the system in English and French with a training corpus of 500 million words for each language, then performed human evaluations in both languages. The results show that the system can generate plausible poetry with high Fluency (3.64) and Coherence (3.41) scores as well as with Meaningfulness (3.27) and Poeticness (3.86).

Bena and Kalita [31] proposed a novel method for generating English poems. They fine-tuned a pre-trained language model GPT-2 [32] to generate poems that express emotion in readers and dream poetry. They classified emotional poems and dreamed text to impact automatic natural-language production in creating poetry. To accomplish this job, they created a meaning for emotion-eliciting material using a word-level emotion lexicon, which was subsequently utilized for training different GPT-2 models. The authors pre-trained the OpenAI-released GPT-2 model on a dataset of first-person dream narratives to teach the network the language of poems. The model was then fine-tuned using a dataset of 20,000 dreams. They rate the proposed model on three qualities: Quality1 (The poem is generally a first-person expression), Quality 2 (The primary substance of the poetry is a dream or vision) and Quality 3 (The poem tells or predicts an experience or event). With scores no lower than 3.2 on the Likert scale for all three qualities, the human evaluation demonstrates that the fine-tuned GPT-2 performed well in generating dream poems [31].

In [33], the authors present an LSTM-based model for generating Persian poems. They trained the model on a dataset of Ghazaliat-e-Hafez and Ghazaliat-e-Saadi[1]. One challenge in using machine learning to generate Persian poetry is the complex grammar of the language. Persian language grammar includes

[1] The Mohammad Qazvini/Gha¯sem Ghani 1941 edition.

various inflections and declensions that can alter word form and meaning. To tackle this challenge, the authors pre-processed and cleaned the dataset before feeding it into the model. Then, they trained the model to predict the following word in a sequence based on the context of the preceding words. The authors conclude that training the model for a larger number of epochs leads to improved poem quality. However, no significant evaluation was presented.

Talafha and Rekabdar [34]-[35] were among the first to apply deep learning to generate Arabic poems. In [34], they proposed a two-phased approach to generate Arabic poems. In the first phase, they generate the first line of the poem by utilizing Bi-GRU (Bi-directional Gated Recurrent Unit) model. In the second phase, they use a modified Bi-GRU encoder-decoder model with hierarchical neural attention to produce the following lines of the poem. In a more recent work [35], Talafha and Rekabdar propose a poetry-generation model with enhanced phonetic and semantic embeddings. In this work, they propose a three-phased approach. First, they use a word-embedding model that represents verses' rhyme and rhythm besides the context. These embeddings offer information on each word's phonetics and its vectorized word representation. Second, they extract N keywords representing the sub-themes of the N verses to generate, where each keyword is to generate one new verse. In the third phase, they use a Bi-GRU encoder-decoder model with word-level and verse-level attention to generate the first verse. Then, they use a hierarchical sequence-to-sequence model to produce the next verses. This last work approach assumes that the number of the extracted keywords is equal to the number of verses to generate. If the input is too short, it will be impossible to generate enough verses. The authors in both works evaluated the generated poetry using BLEU scores and human review. According to human evaluation, their models produced fair-quality poetry.

Recently, Hakami et al. [36] studied using the GPT-2 model to generate Arabic poems. The authors fine-tuned the GPT-2 model, which had already been pre-trained on English corpora. The fine-tuning dataset included 34,466 Arabic verses acquired manually from aldiwan website[2]. In terms of the BLEU-1 score (0.56) and human evaluation (0.5 in Meaning and Coherence), the generated GPT-2 model performed poorly.

Despite the poor results obtained in [36], the effectiveness of the GPT-2-based model in the case of English language poem generation motivated us in [19] to build a GPT-2 model for Arabic poem generation. In the following, we compare the fine-tuned GPT-2-based model to the GPT-3 model and fine-tuned GPT-3 model in Arabic poem generation.

3. GENERATIVE PRE-TRAINED TRANSFORMER

Word-embedding techniques, such as word2vec [37], are able to capture the semantic meaning of words ignoring how the semantics varies across linguistic contexts. However, language models provide a contextualized embedding that improves the representational power of word embeddings. In language modeling, as a first step, a neural network is trained on a large non-annotated corpus of text. During the training, the neural network learns how to recover masked (missing) words or produce the next words. In the second step, we keep the first layers of the neural network and train again the weights of the lasting layers on a smaller specific dataset. These two phases are called the pre-training and fine-tuning phases.

In masked language modeling, like BERT [23], the neural network fills a sub-set of masked words based on all others. This technique is not suitable for text-generation tasks where the network produces the next words based on the previous sequence of words. OpenAI's Generative Pre-trained Transformer (GPT) proposes to pre-train on the standard task of language modeling: predicting the next word in the sequence. GPT models are efficient in text-generation tasks, such as summarization or producing text based on a prompt. In this section, we introduce the three variants of GPT models.

3.1 GPT-1

Radford et al. proposed in [38] the first variant of Generative Pre-trained Transformer (GPT-1) that achieved state- of-the-art results in 9 out of 12 NLP tasks. They employed a semi-supervised learning approach using unsupervised pre-training and supervised fine-tuning.

[2] https://www.aldiwan.net

First, a language-modeling objective is used to maximize the following likelihood given an unsupervised corpus of tokens U = {u_1,..., u_n} to learn the parameters of a neural network:

$$L_1(U) = \sum_i logP(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta) \quad (1)$$

where k is the context window size and the conditional probability P is represented by a neural network with parameters Θ. Stochastic gradient descent [39] is used to train these parameters.

Second, a supervised objective adapts the learned parameters to a particular target task. This objective aims to maximize the likelihood of a given labeled dataset D, where each instance comprises a sequence of input tokens, $x^1, \ldots, x^m$, along with a label y:

$$L_2(D) = \sum_{(x,y)} logP(y \mid x^1, \ldots, x^m) \quad (2)$$

Radford et al. also discovered that adding language modeling as an auxiliary objective to fine-tuning improves learning by enhancing the generalization of the supervised model and accelerates convergence. Therefore, they specifically optimize the following objective with a weight λ (set to 0.5):

$$L_3(D) = L_2(D) + \lambda * L_1(D) \quad (3)$$

The GPT-1 model is recognized as "task agnostic", since it is not limited to a single NLP task, but provides a generalizable architecture that can be adapted to multiple NLP tasks with few adjustments. In terms of architecture, GPT-1 employs a transformer architecture based on a 12-layer decoder (without a decoder) and a self-attention mechanism (12 attention heads). GPT-1 is trained on the BooksCorpus dataset [40] that holds over 7000 unpublished books.

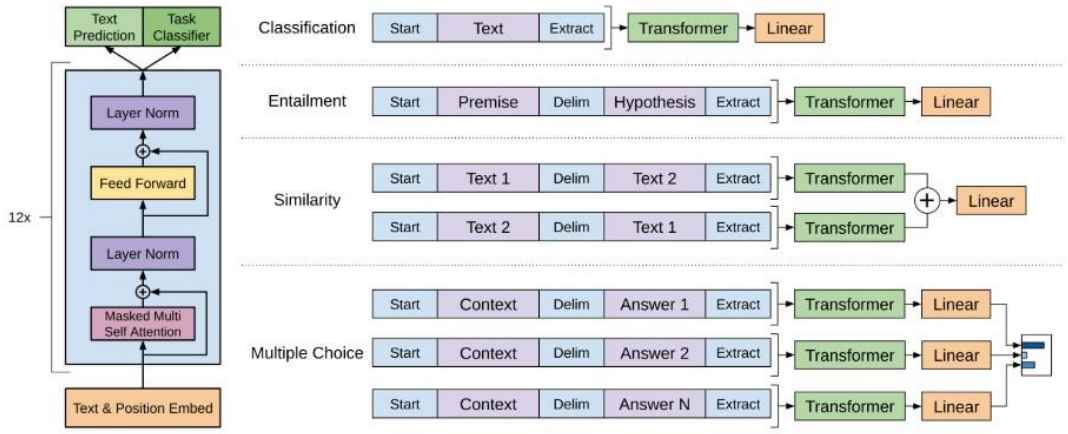


Figure 1. (left) GPT-1 architecture and (right) input transformations for supervised fine-tuning on various tasks [38].

As illustrated in Figure 1, GPT-1 architecture is similar to the decoder-only transformer in [29]. In the network, input tokens $U = (u_{-k}, \ldots, u_{-1})$ are processed through W_e, a token embedding matrix. The activities are then routed through a stack of decoder blocks composed of a multi-headed self-attention layer, a position-wise feedforward layer and a normalization layer:

$$h_0 = UW_e + W_p$$
$$h_l = decoderblock(h_{l-1}) \; \forall i \in [1, n]$$
$$P(u) = softmax(h_n W_e^T)$$

where n is the number of layers and W_p is the position embedding matrix.

Table 2 shows experimental details of the unsupervised pre-training and supervised fine-tuning phases.

Table 2. Hyper-parameters used in GPT-1.

Hyper-parameter	Unsupervised pre-training	Supervised fine-tuning
Max. sequence length	512	512
Batch size	64	32
Learning rate	$2e^{-4}$	$6.25e^{-5}$
# Epochs	100	3

3.2 GPT-2

GPT-2 (Generative Pre-trained Transformer 2) is an unsupervised transformer-based generative language model created by OpenAI [32]. A language model is a machine-learning model that predicts the next word in a given sentence using probability distributions. Using unsupervised methods, language models build many characteristics representing spelling and grammar norms. Unsupervised learning approaches look for patterns in a dataset rather than attempting to find a relationship between the data. GPT-2 was trained using a large corpus (WebText) containing 40 GB of text [32]. This model uses BPE (Byte-Pair Encoding) for encoding text as a sequence of tokens [41]. BPE encoding is a type of sub-word encoding that exists between the character and word levels. Typically, the most common pairs of consecutive bytes of data are encoded as single tokens. However, rare pairs will be encoded as sequences of tokens. This way of encoding catches the recurrent sub-words with specific meanings, like the superlative suffix 'est' in biggest, oldest, …etc. In the Arabic language, suffixes are more common. Arabic suffixes are used as attachable pronouns like تا, which means dual female (they).

With a few structural changes, the GPT-2 architecture closely resembles the GPT-1 model. GPT-2 adds additional layer normalization after the final self-attention block, moves layer normalization to the input of each sub-block and raises context size from 512 to 1024 tokens. Table 3 summarizes the hyper-parameters used to build the various GPT-2 models.

The GPT-2 architecture has demonstrated its ability to represent the English language and has achieved state-of-the- art tasks, such as machine translation, summarization and question-answering. This model is available in four different sizes: small (117 million parameters), medium (345 million parameters), large (762 million parameters) and extra-large (1.5 billion parameters).

Achievements of the GPT-2 model illustrate that training on larger datasets with a larger number of parameters increased the language model's capacity to understand tasks and outperform the state-of-the-art on many tasks in zero-shot. Furthermore, according to the research, as the model's capacity expanded, so did its performance in a log-linear pattern [32].

Table 3. Hyper-parameters used for the four GPT-2 models [32].

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

3.3 GPT-3

In 2020, OpenAI released Generative Pre-trained Transformer 3 (GPT-3) [18]. GPT3 is transformer-based and has the same architecture as GPT-2. More specifically, GPT-3 uses the same updated initialization and pre-normalization used by GPT-2. In addition, GPT-3 architecture includes 96 layers with 96 attention heads in each layer. The context window size was increased from 1024 tokens in GPT-2 to 2048 tokens in GPT-3. GPT-3 was trained on a large and diverse dataset of 499 billion tokens, including Common Crawl, web texts, books and Wikipedia (45TB of text data). With 175 billion parameters, GPT-3 was 10 times larger than any previous language model and had 100 times more parameters than GPT-2. GPT-3 is proposed to be used for all NLP tasks without the need for gradient updates or fine-tuning. GPT-3 performs well on various NLP tasks, including translation, question-answering and cloze tasks [18]. GPT-3 is available in eight versions with a different number of trainable parameters, as listed in Table 4.

GPT-3 shows strong performance on many NLP tasks and benchmarks in the zero-shot, one-shot and few-shot settings, in some cases nearly matching the performance of state-of-the-art fine-tuned systems.

GPT-3 was not made available; instead, access was to be allowed *via* an API, giving the model's developers more control over its use. At the time of writing, the API was under beta testing. However, the API is typically used to start the model by providing a prompt and some introductory text.

Table 4. Sizes, architectures and learning hyper-parameters of the GPT-3 models [42].

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3 B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7 B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7 B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0 B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0 B	96	12288	96	128	3.2M	0.6×10^{-4}

4. PROPOSED MODELS FOR POEM GENERATION

In this section, we describe the pre-training data, the training setup and the fine-tuning setup that we used to build a GPT-2 model for poem generation. We also report on the setup that we used to fine-tune two variants of the GPT-3 model.

4.1 GPT-2 Based Model

In our previous work [19], we proposed a fine-tuned GPT-2 model capable of generating Arabic poetry. However, GPT-2 is only trained in English, while GTP-3 is trained in several languages, including Arabic. To build a GPT-2 model, we went through the two phases of the training process: pre-training and fine-tuning. In the pre-training phase, we used two publicly available corpora Khaleej-2004 [43] and Watan-2004 [44]. Khaleej-2004 is an MSA (Modern Standard Arabic) corpus collected from thousands of articles downloaded from Akhbar Al Khaleej, an online newspaper. The corpus contains 5,690 documents, totaling more than 2 million words. Watan-2004 is another MSA corpus composed of nearly 20,000 documents containing more than 9 million words. To fine-tune our model, we used the Arabic poetry dataset[3] that was scrapped completely from aldiwan website[4]. The Arabic poetry dataset has 55K poems for over 540 poets from 9 different eras. Table 5 summarizes the number of words and unique words for each dataset used in building our fine-tuned GPT-2 model.

After collecting the data, we had to pre-train our model on Arabic corpora. We used the small versions of the pre- trained GPT-2 Tokenizer and Model from the Transformers Library (Hugging Face[5]). This Library provided us with the tokenizer structure needed as well as with pre-trained model weights. Rather than starting with a random network, we trained our GPT-2 model in Arabic with weights already trained in English. Next, we trained a BPE tokenizer on the Arabic corpus using the Tokenizers' Library (Hugging Face), where we obtained a vocabulary of 50K tokens. Then, we used Google Colab to pre-train our GPT-2 model on the Khaleej-2004 and Watan-2004 corpora. The pre-training was executed on an NVIDIA Tesla T4 (16 GB) GPU for 25 epochs. The pre-training lasted about 32 hours.

For the Arabic poem-generation task, we fine-trained our pre-trained model on poems from the Arabic poetry dataset. We used the same GPU (used in the pre-training stage) for 6 epochs to fine-tune our model. The fine- tuning lasted 12 hours. Table 6 shows the hyper-parameter values used in the pre-training and the fine-tuning steps.

Table 5. Summary of the used datasets [19].

Dataset	#Words	#Unique Words
Khaleej-2004	2.482K	122K
Watan-2004	9.813K	291K
Total pre-training	12.229K	413K
Arabic poetry	6.933K	2.060K

Table 6. Hyper-parameters used to build a GPT-2 Arabic poem generator [19].

Hyper-parameter	Pre-training	Fine-tuning
Max. sequence length	1024	1024
Batch size	24	24
Learning rate	$3e^{-5}$	$3e^{-5}$
# Epochs	25	6

[3] https://www.kaggle.com/ahmedabelal/arabic-poetry
[4] https://www.aldiwan.net
[5] https://huggingface.co/

4.2 GPT-3 Based Models

Since GPT-3 is pre-trained on a corpus containing Arabic language and we are not able to pre-train such a big model, we decided to fine-tune the GPT-3 models without the pre-training phase. Through GPT-3 API, OpenAI offers the possibility to build new models by fine-tuning GPT-3 models. We used the provided API to fine-tune GPT-3 on Arabic poems. OpenAI has publicly released four versions of GPT-3: Ada, Babbage, Curie and Davinci. The fastest model is Ada, while the most capable model is Davinci. For reasons of computation cost, we fine-tuned only two versions of GPT-3: Ada and Davinci models. We fine-tuned the Ada and Davinci models during 4 epochs on 5,223 poems and 114 poems from an Arabic poetry dataset, respectively. More precisely, we fine-tuned the Ada model on 10% of the data used to fine-tune GPT-2. For the Davinci model, we used 0.2% of the Arabic poetry dataset. This fine-tuning is considered as a few-shot fine-tuning, as claimed by GPT-3 authors [18]. Few-shot fine-tuning is a method of adapting a pre-trained model to a new task using a small amount of labeled data [18]. It involves updating the weights of the pre-trained model on the new task, poem-generation task in our case. This technique is often used when it is difficult or expensive to fine-tune the model on a big amount of data. In Table 7, we detail the hyper-parameter values used in this fine-tuning.

Table 7. Proposed models' fine-tuning configurations.

Models	Max. seq. len.	Batch size	Learning rate	# Epochs	# Poems	Model size
GPT-2	1024	24	$3e^{-5}$	6	55,000	117M
GPT-3 Ada	2048	64	0.1	4	5,223	2.7B
GPT-3 Davinci	4000	64	0.1	4	114	175B

5. EVALUATION

We evaluate fine-tuned GPT-2, GPT-3 Davinci, fine-tuned GPT-3 Ada and fine-tuned GPT-3 Davinci on the task of poem generation based on a set of two input verses. Each model is asked to generate the N following verses of the two verses given as input. To build the evaluation inputs, we randomly picked five Arabic poems not included in the fine-tuning dataset. From each poem, we extract two verses to be used as input. Table 8 displays the five Arabic verses used as evaluation inputs.

Table 8. Arabic verses used as evaluation inputs.

Input verses' samples	**Input verses samples in English**
أنا الذي نظر الأعمى إلى أدبي وأسمعت كلماتي من به صمم أنام ملء جفوني عن شواردها ويسهرُ الخلقُ جراها ويختصم	I am the one whose verse is seen (even) by the blind and whose words are heard (even) by the deaf I enjoy my sweet repose, not concerning myself with poetry, whereas others burn the midnight oil, in endless literary disputes
أراك عصي الدمع شيمتك الصبر أما للهوى نهي عليك ولا أمر؟ بلى أنا مشتاق وعندي لوعة ولكن مثلي لا يذاع له سر	I see you holding back the tears, your habit is patience Does not love has on you a prohibition or an order? Yes, I miss with a burning desire, but someone like me doesn't spread secrets
لا يكتم السر إلا من له شرف والسر عند كرام الناس مكتوم السر عندي في بيت له غلق ضلت مفاتيحه والباب مردوم	No one keeps a secret except those with honor and the secret by generous people is kept The secret to me is in a house that has a lock its keys are lost and the door is buried
قالت غلبتك يا هذا فقلت لها لم تغلبيني ولكن زدتني كرماً بعض المعارك في خسرانها شرف من عاد منتصراً منها أو انهزما	She said I have beaten you, so I told her you didn't beat me, but you gave me more generousity To loose some battles is an honor for both who returned as winners and those who were losers
في مدخل الحمراء كان لقاؤنا ما أطيب اللقيا بلا ميعاد عينان سوداوان في حجريها تتوالد الأبعاد من أبعاد	At the entrance of Alhambra we have met how delightful it is to meet without a rendezvous Two dark eyes: in their depths distances give birth to distances

For each input, each model is asked to generate the N following verses, with N taking the values of 2, 4 and 6. The quality of the generated verses will be evaluated regarding the meaning and the rhyme of the verses fed as inputs.

To generate Arabic poems using GPT-3 Davinci, fine-tuned GPT-3 Ada and fine-tuned GPT-3 Davinci, we use the OpenAI Beta Playground[6]. The Playground is a web-based application that allows users to quickly test the prompts and get familiar with how the API works. We use BLEU scores and human evaluation to evaluate the verses generated by each model. In the following, we introduce both evaluation methods.

5.1 BLEU Scores

BLEU (Bilingual Evaluation Understudy) scores [45] are commonly used in machine translation (MT) to compare reference and candidate texts. The BLEU-1, BLEU-2, BLEU-3 and BLEU-4 represent the number of unigrams, bigrams, trigrams and 4-grams, respectively. Each gram reflects the number of words selected from both texts and compared to one another. A BLEU score has a value ranging from 0 to 1. In earlier studies, including [46][47][48], BLEU scores were also used to evaluate poem generation. A better generated poem usually achieves a higher BLEU score, as it shares more n-grams with the referenced poem. Therefore, we use BLEU scores to evaluate the verses generated by the fine-tuned GPT-2, GPT-3 Davinci, fine-tuned GPT-3 Ada and fine-tuned GPT-3 Davinci models. For the poems generated by the four models in two, four and six verses, we calculated the BLEU-1, BLEU-2, BLEU-3 and BLEU-4 scores. We used test data from the Arabic poetry dataset containing 7,136 verses as a reference to compute these scores.

5.2 Human Evaluation

To evaluate the quality of generated poems, we performed a human evaluation following previous works [19], [46], [34]. We asked three human experts to judge 240 verses generated by fine-tuned GPT-2, GPT-3 Davinci, fine-tuned GPT-3 Ada and fine-tuned GPT-3 Davinci models. This evaluation is based on four criteria: Fluency, Coherence, Meaning and Poeticness. Fluency checks whether the generated poem is grammatically satisfactory. Coherence inspects whether the generated poem is thematically coherent. Meaning measures how meaningful the content of a generated poem is. Poeticness estimates the features of poetry in the generated poem. These four criteria are rated by the human evaluators on a scale ranging from 1 (worst) to 5 (best).

Table 9. Inter-annotator agreement.

Variable	Krippendorff's α
Fluency	0.90
Coherence	0.81
Meaning	0.79
Poeticness	0.85

We employ Krippendorff's α [49] Inter-Annotator Agreement (IAA) to estimate annotation reliability. Krippendorff's α is predicated on the notion that predicted agreement is computed by looking at the total distribution of ratings, regardless of who issued them. Table 9 shows Krippendorff's α calculated for each dimension. Table 9 indicates that the reliabilities ranged between 0.90 and 0.79, showing that the annotators' judgments were consistent.

6. EXPERIMENTS

We first evaluated our fine-tuned GPT-2 model against the state-of-the-art models for Arabic poem generation: Vanilla RNN, LSTM, GRU, RNN EncoderDecoder (with and without attention) and Bi-GRU with hierarchical neural attention. Then, we evaluated our model against GPT-3 models: GPT-3 Davinci, fine-tuned GPT-3 Davinci and fine-tuned GPT-3 Ada. The results are detailed and discussed in the following sub-sections.

6.1 GPT-2 against State-of-the-art Models

6.1.1 BLEU Scores

We evaluate our fine-tuned GPT-2 results of BLEU scores in generating two verses against the work

[6] https://beta.openai.com/playground

of Talafha and Rekabdar in [34]-[35]. The fine-tuned GPT-2 model is better than other state-of-the-art models for BLEU-1, BLEU-2, BLEU-3 and BLEU-4, as shown in Table 10.

We observe that all BLEU scores are low. Since there are multiple ways to compose a poem given two verses (see Appendix A), we believe it is rational to obtain low BLEU scores. This observation can be confirmed by the scores decreasing from BLEU-1 to BLEU-4. The probability of obtaining shared n-grams between the generated poem and the reference poem (BLEU-n) decreases when the number of words (n) composing the n-grams increases.

Table 10. BLEU comparison with Talafha and Rekabdar's [34]-[35] work.

Models	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**
Vanilla RNN	0.0211	0.0199	0	0
LSTM	0.1522	0.1124	0.0081	0.0013
GRU	0.1512	0.1139	0.0084	0.0021
RNN EncoderDecoder (without attention)	0.2513	0.1539	0.0740	0.0510
RNN EncoderDecoder (with attention)	0.3010	0.2110	0.0911	0.0801
Bi-GRU with hierarchical neural attention [34]	0.4122	0.3144	0.204	0.1092
Phonetic CNN_sub-word embedding model [35]	0.5301	0.4010	0.3001	0.1500
Fine-tuned GPT-2	**0.8726**	**0.5572**	**0.3418**	**0.2001**

6.1.2 Human Evaluation

We also compared the fine-tuned GPT-2 results of the human evaluation in generating two verses with the work of Talafha and Rekabdar [34]-[35]. Table 11 reports the results of this comparison. Results in Table 11 show that in terms of Poeticness and Fluency, the fine-tuned GPT-2 model outperforms the other models. In terms of Coherence and Meaning, the fine-tuned GPT-2 model obtains acceptable results compared to the other models. This can be explained by the fact that Talafha and Rekabdar's work is focused on specific topics: love and religion. Contrasting our work, the covered topics are multiple and include most of the Arabic poetry topics. We also recall that in Talafha and Rekabdar's work, each verse is generated from a keyword. In comparison, our model is only constrained by the verses in inputs.

Table 11. Fine-tuned GPT-2 compared to the work of Talafha and Rekabdar [34]-[35].

Models	**Fluency**	**Coherence**	**Meaning**	**Poeticness**
Vanilla RNN	0.1	0.8	0.7	0
LSTM	0.3	0.9	0.8	0.1
GRU	0.3	1.0	1.0	0.2
RNN Encoder- Decoder (without attention)	2.0	1.5	2.4	0.3
RNN Encoder- Decoder (with attention)	2.3	2.5	2.7	0.4
Bi-GRU with hierarchical neural attention [34]	2.1	3.2	3.5	0.9
Phonetic CNN_sub-word embedding model [35]	2.7	**3.3**	**3.6**	2.5
Fine-tuned GPT-2	**3.2**	2.8	2.4	**4.0**

6.2 GPT-2 against GPT-3 Models

6.2.1 BLEU Scores

Table 12 shows that the fine-tuned GPT-2 model outperformed the GPT-3 Davinci, the fine-tuned GPT-3 Ada and the fine-tuned GPT-3 Davinci models for BLEU scores in generating two, four and six Arabic poem verses. We think that fine-tuning on poem dataset forces the model to generate text closer to the poetic context. The fine-tuned GPT-3 Davinci model achieved better BLEU scores than the raw GPT-3 Davinci model. We also observe that GPT-3 Davinci obtains better BLEU scores than fine-tuned GPT-3 Ada when generating 2 verses, while fine-tuned GPT-3 Ada performs better in 4- and 6-verse generation. These observations show that a fine-tuned smaller model has a better generation performance than a non-fine-tuned larger model.

Table 12. The BLEU scores of different GPT models.

Models	BLEU-1	BLEU-2	BLEU-3	BLEU-4
2-verse generation				
GPT-3 Davinci	0.6846	0.4066	0.2404	0.1375
Fine-tuned GPT-2	**0.8726**	**0.5572**	**0.3418**	**0.2001**
Fine-tuned GPT-3 Ada	0.6668	0.3903	0.2298	0.1310
Fine-tuned GPT-3 Davinci	0.7075	0.4200	0.2515	0.1449
4-verse generation				
GPT-3 Davinci	0.6548	0.3853	0.2272	0.1297
Fine-tuned GPT-2	**0.8456**	**0.5151**	**0.3064**	**0.1764**
Fine-tuned GPT-3 Ada	0.7103	0.4248	0.2532	0.1455
Fine-tuned GPT-3 Davinci	0.7288	0.4374	0.2611	0.1502
6-verse generation				
GPT-3 Davinci	0.5648	0.3278	0.1927	0.1095
Fine-tuned GPT-2	**0.8211**	**0.5123**	**0.3095**	**0.1796**
Fine-tuned GPT-3 Ada	0.7060	0.4188	0.2485	0.1424
Fine-tuned GPT-3 Davinvi	0.7200	0.4336	0.2586	0.1486

6.2.2 Human Evaluation

Automatic evaluation metrics like BLEU scores are fast and cost-effective measurements of the quality of poem-generation models. However, as poetry is an art form subject of human appreciation, human judgment is the benchmark to assess the quality of the generated poems. The results of the human evaluation of GPT-2 and other GPT-3 models are listed in Table 13.

Table 13. The results of human evaluation of different GPT models.

Models	Fluency	Coherence	Meaning	Poeticness
2-verse generation				
GPT-3 Davinci	2.7	2.1	2.3	1.4
Fine-tuned GPT-2	3.2	**2.8**	2.4	**4.0**
Fine-tuned GPT-3 Ada	3.1	2.4	2.5	2.3
Fine-tuned GPT-3 Davinci	**4.0**	2.5	**3.0**	2.5
4-verse generation				
GPT-3 Davinci	2.9	**2.4**	2.4	1.7
Fine-tuned GPT-2	3.0	2.1	2.3	**3.5**
Fine-tuned GPT-3 Ada	3.0	2.2	2.2	2.4
Fine-tuned GPT-3 Davinci	**3.5**	2.2	**2.5**	2.3
6-verse generation				
GPT-3 Davinci	3.5	2.1	1.9	1.9
Fine-tuned GPT-2	3.4	2.0	1.9	**3.7**
Fine-tuned GPT-3 Ada	3.4	2.3	2.5	2.7
Fine-tuned GPT-3 Davinci	**3.6**	**2.4**	**2.6**	2.1

Results in Table 13 show that the fine-tuned GPT-2 scored higher on Poeticness than GPT-3 and fine-tuned GPT-3 in generating two, four and six Arabic poem verses. Regarding Fluency, Coherence and Meaning, the four models performed similarly. We also notice from Table 13 that the two fine-tuned GPT-3 models get a better result in Poeticness than GPT-3 without fine-tuning when generating four and six verses. The GPT-3 Davinci (without fine-tuning) has the lowest score in Poeticness compared to the other models. On the other hand, the fine- tuned GPT-2 model performance degrades in Coherence and Meaning, but maintains good performance in terms of Fluency and Poeticness when generating more verses. Albeit, in the fine-tuned GPT-3 Davinci, we find that Poeticness decreases when generating more verses.

Human-evaluation results show that fine-tuning GPT-3 Davinci model improved its Poeticness

capabilities, since it achieved better Poeticness scores than the raw GPT-3 Davinci model. We also observe that fine-tuning improved GPT-3 Davinci Meaning capabilities, maintaining similar Fluency and Meaning scores. In addition, the smallest model, fine-tuned GPT-2, obtained the best Poeticness scores in all runs. Human-evaluation observations agree with the findings of the automatic evaluation (BLEU scores). These observations show that the size of the model is less significant than fine-tuning in the case of automatic poem generation.

Table 14 presents a sample of outputs generated by the four models given the same input. The two verses used as input belong to the Kamil meter (بحر الكامل). The usual form of this meter is as follows:
| uu – u – | uu – u – | uu – u – |
where "–" represents a long syllable, "u" a short syllable and "uu" one long or two short syllables.

Table 14. Examples of verses generated by the different GPT models.

Input		
	Arabic verses	English translation
	في مدخل الحمراء كان لقاؤنا ما أطيب اللقيا بلا ميعاد عينان سوداوان في حجريهما تتوالد الأبعاد من أبعاد	At the entrance of Alhambra we have met how delightful it is to meet without a rendezvous Two dark eyes: in their depths distances give birth to distances
Outputs		
Model	Arabic verses	English translation
Fine-tuned GPT-2	قد كان في البحر البهيم مسلها بالركب حين تكون للأوراد يا ليل يا ليل انتبهوا على ما كان في بغداد من أعراس	It was in the rough sea, a place on the way, when you are for the source of water Oh night, oh night, pay attention to what was in Baghdad weddings
GPT-3 Davinci	ليس مع فكرة الصداقة شيء غريب ما اطيب اللقيا بلا ميعاد دايخة الحرب بين الأحزان والأغراب الحزن في الحرب إذا ما فتح الصدر	There is nothing strange about the idea of friendship how delightful it is to meet without a rendezvous The bitterness of the war between the sorrows and the strangers sadness in war is when the chest is opened
Fine-tuned GPT-3 Ada	في حمراء حفاظن كما أطيبهم وفي شرب بدران أكثار حراد في حمراء ريحانات تتوالد في حسنات حميانات تتوالد	In Hamra, we have preserves because they are the best and in Badran drinking more Harad In Hamra, Rihanat are born In good deeds, fevers are born
Fine-tuned GPT-3 Davinci	أحبابي وأصدقائي وإخواني أشوفهم وأنا أسمع منهم أصوات أحبابي المنادية وأنا أعيش في دار الحمراء	My lovers, my friends and my brothers I see them and I hear them The calling voices of my beloved and I live in the Hamra house

The example generated by the fine-tuned GPT-2 model (see Table 14) perfectly maintains the same meter of the input, while the other models do not maintain this meter. We notice that the fine-tuned GPT-3 Davinci model generated better rhyme than GPT-3 Davinci model. We also observe that except GPT-3 Davinci model, all the models generated love verses. This capability to generate the same thematic verses is enabled during the fine-tuning phase. Another remarkable fact is the reuse of the word "Hamra" ("الحمراء") by the two fine-tuned GPT-3 models. In free text generation, the model generates the next word based on the previous context (input text). In such a case, the model effort focuses on optimizing the meaning of the generated text. However, in the context of poem-generation, the model effort focuses on the rhyme, the thematic and the meaning of the previous context (input verses, in poetry). The fine-tuning phase is responsible for adapting the model to the underlying task. This is confirmed by the observations on the generated verses illustrated in Table 14. We observed that with fewer parameters and well defined fine-tuning, our fine-tuned GPT-2 model outperformed larger models with less fine-tuning in the task of Arabic poem generation.

7. CONCLUSION

In this paper, we tackled an automatic Arabic poem-generation task. We competed for different deep neural network architectures to emphasize the significance of the model size and the fine-tuning phase in the case of Arabic poem generation. We showed in the state-of-the-art review that the Generative

Pre-trained Transformer (GPT) architecture fits best for automatic poem generation. We presented and compared four models for automatic Arabic poem generation: fine-tuned GPT-2, raw GPT-3 Davinci, fine-tuned GPT-3 Ada and fine-tuned GPT-3 Davinci. We proposed to evaluate these four models on two, four and six verses of Arabic poem generation. We used BLEU scores and human evaluation to assess the quality of the poems generated by the four models.

Experiments revealed that the poeticness capability strongly depends on the quality of fine-tuning phase, even for very large models. The fine-tuned GPT-2 model measured higher BLEU scores than all the GPT-3 models. The human evaluation shows that the smaller version of GPT-2 fine-tuned on Arabic poems outperforms the most capable version of GPT-3 Davinci in the quality of poem generation. Human evaluation also shows that Ada, the minor publicly available version of GPT-3 fine-tuned on a small dataset, performs better than GPT-3 Davinci without fine-tuning in terms of Poeticness, which is the most critical criterion in poem generation.

We conclude that the size of the model is less significant than fine-tuning in the case of automatic poem generation. Thereafter, researchers with limited computation resources should not be discouraged by the size of the latest models. A trade-off between the model size and its text generation performance is still possible through fine- tuning, which is less resource-intensive. Nevertheless, it must not be forgotten that other architectures exist. In future work, we plan to explore other deep neural network architectures to improve the quality of the generated Arabic poems. However, many challenges need to be addressed to achieve effectiveness. We also plan to focus on particular topics of Arabic poems to enhance the Coherence and the Meaning of the generated poems.

REFERENCES

[1] S. Subramanian, S. Rajeswar, F. Dutil, C. Pal and A. Courville, "Adversarial Generation of Natural Language," Proc. of the 2nd Workshop on Representation Learning for NLP, pp. 241–251, 2017.

[2] R. Yan, H. Jiang, M. Lapata, S.-D. Lin, X. Lv and X. Li, "i, Poet: Automatic Chinese Poetry Composition through a Generative Summarization Framework under Constrained Optimization," Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence, pp. 2197-2203, 2013.

[3] A. Das and B. Gambäck, "Poetic Machine: Computational Creativity for Automatic Poetry Generation in Bengali," Proc. of the Int. Conf. on Innovative Comp. and Cloud Comp. (ICCC), pp. 230–238, 2014.

[4] H. G. Oliveira and A. Cardoso, "Poetry Generation with PoeTryMe," Proc. of Computational Creativity Research: Towards Creative Machines, Part of the Atlantis Thinking Machines Book Series (ATLANTISTM), vol. 7, pp. 243–266, Springer, 2015.

[5] M. Ghazvininejad, X. Shi, Y. Choi and K. Knight, "Generating Topical Poetry," Proc. of the 2016 Conf. on Empirical Methods in Natural Language Processing, pp. 1183–1191, Austin, Texas, 2016.

[6] M. Ghazvininejad, X. Shi, J. Priyadarshi and K. Knight, "Hafez: An Interactive Poetry Generation System," Proc. of ACL 2017, System Demonstrations, pp. 43–48, Vancouver, Canada, 2017.

[7] D. Singh, M. Ackerman and R. Y. Pérez, "A Ballad of the Mexicas: Automated Lyrical Narrative Writing," Proc. of the 8ht Int. Conf. on Computational Creativity (ICCC), [Online], Available: http://ilitia.cua.uam.mx:8080/jspui/handle/123456789/442, 2017.

[8] L. Xu, L. Jiang, C. Qin, Z. Wang and D. Du, "How Images Inspire Poems: Generating Classical Chinese Poetry from Images with Memory Networks," Proc. of the 32nd Conf. on Artificial Intelligence, vol. 32, Article No. 689, pp. 5618–5625, 2018.

[9] A. Zugarini, S. Melacci and M. Maggini, "Neural Poetry: Learning to Generate Poems Using Syllables," Proc. of the Int. Conf. on Artificial Neural Networks, pp. 313–325, DOI: 10.1007/978-3-030-30490-4_26, Springer, 2019.

[10] T. Van de Cruys, "Automatic Poetry Generation from Prosaic Text," Proc. of the 58th Annual Meeting of the Associ. for Computational Linguistics, pp. 2471–2480, DOI: 10.18653/v1/2020.acl-main.223, 2020.

[11] C.-L. Zhou, W. You and X. Ding, "Genetic Algorithm and Its Implementation of Automatic Generation of Chinese Songci," Journal of Software, vol. 21, no. 3, pp. 427–437, 2010.

[12] J. He, M. Zhou and L. Jiang, "Generating Chinese Classical Poems with Statistical Machine Translation Models," Proc. of the 26th AAAI Conference on Artificial Intelligence, vol. 26, no. 1, pp. 1650–1656, DOI: 10.1609/aaai.v26i1.8344, 2012.

[13] Q. Wang, T. Luo and D. Wang, "Can Machine Generate Traditional Chinese poetry? A Feigenbaum Test," Proc. of the Int. Conf. on Brain Inspired Cognitive Systems, Part of the Lecture Notes in Computer Science Book Series (LNAI), vol. 10023, pp. 34–46, Springer, 2016.

[14] J. Zhang, Y. Feng, D. Wang, Y. Wang, A. Abel, S. Zhang and A. Zhang, "Flexible and Creative Chinese Poetry Generation Using Neural Memory," arXiv: 1705.03773, DOI: 10.48550/arXiv.1705.03773, 2017.

[15] Z. Wang, W. He, H. Wu, H. Wu, W. Li, H. Wang and E. Chen, "Chinese Poetry Generation with Planning

Based Neural Network," arXiv:1610.09889, DOI: 10.48550/arXiv.1610.09889, 2016.

[16] X. Yang, X. Lin, S. Suo and M. Li, "Generating Thematic Chinese Poetry Using Conditional Variational Auto-encoders with hybrid decoders," arXiv: 1711.07632, DOI: 10.48550/arXiv.1711.07632, 2017.

[17] Z. N. Abdel-Malek, Towards a New Theory of Arabic Prosody, 5th Edn., ISSN: 0258-3976, Tajdid Online Forum for Facilitating Arabic Studies, 2019.

[18] T. Brown, B. Mann, N. Ryder et al., "Language Models Are Few-shot Learners," Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020.

[19] M. E. G. Beheitt and M. B. H. Hmida, "Automatic Arabic Poem Generation with GPT-2," Proc. of the 14th Int. Conf. on Agents and Artificial Intelligence (ICAART 2022), vol. 2, pp. 366-374, 2022.

[20] X. Yi, M. Sun, R. Li and Z. Yang, "Chinese Poetry Generation with a Working Memory Model," Proc. of the 27th Int. Joint Conference on Artificial Intelligence (IJCAI'18), pp. 4553–4559, 2018.

[21] Z. Liu, Z. Fu, J. Cao, G. de Melo, Y.-C. Tam, C. Niu and J. Zhou, "Rhetorically Controlled Encoder-Decoder for Modern Chinese Poetry Generation," Proc. of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1992–2001, Florence, Italy, 2019.

[22] L. Deng, J. Wang, H. Liang et al., "An Iterative Polishing Framework Based on Quality Aware Masked Language Model for Chinese Poetry Generation," Proc. of the AAAI Conference on Artificial Intelligence, pp. 7643–7650, 2020.

[23] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers), pp. 4171–4186, 2019.

[24] L. Shen, X. Guo and M. Chen, "Compose Like Humans: Jointly Improving the Coherence and Novelty for Modern Chinese Poetry Generation," Proc. of the 2020 IEEE Int. Joint Conf. on Neural Networks (IJCNN), pp. 1–8, Glasgow, UK, 2020.

[25] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv: 1412.6980, 2014.

[26] J. H. Lau, T. Cohn, T. Baldwin, J. Brooke and A. Hammond, "Deep-speare: A Joint Neural Model of Poetic Language, Meter and Rhyme," Proc. of the 56th Annual Meeting of the Association for Computational Linguistics, vol. 1: Long Papers, pp. 1948–1958, 2018.

[27] D. Bahdanau, K. H. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," Proc. of the 3rd Int. Conf. on Learning Representations (ICLR 2015), arXiv: 1409.0473, 2015.

[28] M. C. Santillan and A. P. Azcarraga, "Poem Generation Using Transformers and doc2vec Embeddings," Proc. of the IEEE Int. Joint Conf. on Neural Networks (IJCNN), pp. 1–7, Glasgow, UK, 2022.

[29] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention Is All You Need," arXiv: 1706.03762, DOI: 10.48550/arXiv.1706.03762, 2017.

[30] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," Proc. of the Int. Conf. on Machine Learning (PMLR), pp. 1188–1196, arXiv: 1405.4053, 2014.

[31] B. Bena and J. Kalita, "Introducing Aspects of Creativity in Automatic Poetry Generation," arXiv: 2002.02511, DOI: 10.48550/arXiv.2002.02511, 2020.

[32] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei and I. Sutskever, "Language Models Are Unsupervised Multitask Learners," OpenAI blog, vol. 1, no. 8, p. 9, 2019.

[33] M. H. Moghadam and B. Panahbehagh, "Creating a New Persian Poet Based on Machine Learning," arXiv e-prints, pp. arXiv–1810, DOI: 10.48550/arXiv.1810.06898, 2018.

[34] S. Talafha and B. Rekabdar, "Arabic Poem Generation with Hierarchical Recurrent Attentional Network," Proc. of the IEEE 13th Int. Conf. on Semantic Comp. (ICSC), pp. 316–323, Newport Beach, USA, 2019.

[35] S. Talafha et al., "Poetry Generation Model *via* Deep Learning Incorporating Extended Phonetic and Semantic Embeddings," Proc. of the IEEE 15th Int. Conf. on Semantic Comp., pp. 48–55, USA, 2021.

[36] A. Hakami, R. Alqarni, M. Almutairi and A. Alhothali, "Arabic Poems Generation Using LSTM, Markov-LSTM and Pre-trained GPT-2 Models," Computer Science & Information Technology (CS&IT), vol. 11, no. 15, pp. 139–147, 2021.

[37] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," arXiv:1310.4546, DOI: 10.48550/arXiv.1310.4546 2013.

[38] A. Radford, K. Narasimhan, T. Salimans et al., "Improving Language Understanding by Generative Pre-training," [Online], Available: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[39] H. Robbins and S. Monro, "A Stochastic Approximation Method," The Annals of Mathematical Statistics, vol. 22, no. 3, pp. 400–407, 1951.

[40] Y. Zhu, R. Kiros, R. Zemel et al., "Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books," Proc. of the IEEE Int. Conf. on Computer Vision (ICCV), pp. 19–27, DOI: 10.1109/ICCV.2015.11, 2015.

[41] R. Sennrich, B. Haddow and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," arXiv: 1508.07909, DOI: 10.48550/arXiv.1508.07909, 2015.

[42] T. B. Brown, B. Mann, N. Ryder et al., "Language Models Are Few-shot Learners," arXiv: 2005.14165,

DOI: 10.48550/arXiv.2005.14165, 2020.

[43] M. Abbas and K. Smaili, "Comparison of Topic Identification Methods for Arabic Language," Proc. of Int. Conf. on Recent Advances in Natural Lang. Process. (RANLP), pp. 14–17, Borovets, Bulgaria, 2005.

[44] M. Abbas, K. Smaïli and D. Berkani, "Evaluation of Topic Identification Methods on Arabic Corpora," Journal of Digital Information Management, vol. 9, no. 5, pp. 185–192, 2011.

[45] K. Papineni, S. Roukos, T. Ward and W.-J. Zhu, "BLEU: A Method for Automatic Evaluation of Machine Translation," Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 311–318, Philadelphia, USA, 2002.

[46] J. Li, Y. Song, H. Zhang, D. Chen, S. Shi, D. Zhao and R. Yan, "Generating Classical Chinese Poems *via* Conditional Variational Autoencoder and Adversarial Training," Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing, pp. 3890–3900, Brussels, Belgium, 2018.

[47] R. Yan, "i, Poet: Automatic Poetry Composition through Recurrent Neural Networks with Iterative Polishing Schema," Proc. of the 25th Int. Joint Conf. on Artificial Intell. (IJCAI-16), pp. 2238–2244, 2016.

[48] X. Zhang and M. Lapata, "Chinese Poetry Generation with Recurrent Neural Networks," Proc. of the 2014 Conf. on Empirical Methods in Natural Lang. Process. (EMNLP), pp. 670–680, Doha, Qatar, 2014.

[49] K. Krippendorff, Content Analysis: An Introduction to Its Methodology, 3rd Edition, Sage Publications, 2013.

APPENDEX. MULTIPLE WAYS TO COMPOSE A POEM

This Appendix accompanies our fine-tuned GPT-2 experiments in Section 6. It illustrates the various ways to compose a poem from a given input. More explicitly, as illustrated in Table 15, we feed the model two verses as input and ask the model to generate the two following verses. The generated verses should keep the rhyme scheme of the verses fed as input. Each output listed in Table 15 represents a different model run to generate the two following verses of the input listed in Table 15.

Table 15. Examples of multiple Arabic verses generated given two verses.

	Arabic verses	**English translation**
Input		
	يا منزل الآيات و الفرقان بيني وبينك حُرمة القرآن اشرح به صدري لمعرفة الهدى واعصم به قلبي من الشيطان	O revealer of verses and furqan between me and you is the sanctity of Quran Open my chest with it to know the guidance and protect my heart from the devil
Outputs		
Output 1	يا ليت شعري قد بكى زمن الأسى والقلب في قلبي هوى الإنسان أيام كانت أمّةٌ لكعبة شعرهم فأراهما وهو الرّدى في آن	I wish my poetry had cried on the time of sorrow and the heart in my heart is the love of the human The days when it was a nation for the Kaaba of they poetry and he saw them and he was the one who was at the same time
Output 2	وأملأ جمال القلب في بغداد ما ينجو من الأغصان والرضوان واسلم هديت الشعر من خمر الهوى يا من له ما أنت في الإنسان	And fill the beauty of heart in Baghdad what survives of the branches and the Radwan And thanks I gave the poetry from the wine of love o you who have what you are in human
Output 3	يا سيدي هذا الذي أحيا به قلبي الذي أهواه من إيماني إن كان ظني من سواك فإنني من بعد فقدك في ذرى الأزمان	Sir, this is what I live by my heart that I love from my faith If I think of anyone other than you, then I'm after losing you in the pinnacles of time

ملخص البحث:

تُعدّ كتابة الشِّعر تحدّياً لبني البَشر من حيث حاجتها الى الإبداع والى مستوى عالٍ من القدرة اللغوية، لذا فإنّ التّوليد الأوتوماتيكي للشِّعر مجالاً مهماً للبحث جَذَب اهتمام مجتمع معالجة اللغات الطبيعية. فقد فحص العديد من الباحثين التّوليد الأوتوماتيكي للشِّعر باستخدام طُرُق التّعلُّم العميق، إلا أنّ القليل من تلك البحوث ركّز على الشِّعر العربي.

في هذا البحث، نَعْرض كيفية الاستفادة من نماذج GPT-2 و GPT-3 لتوليد الشِّعر العربي أوتوماتيكياً. وقد تمّ استخدام أربعةٍ من هذه النّماذج ومن تمّ تقييمها آليّاً وبشرياً. وقد أثبتت نتائج التّقييم بِنَوْعَيْهِ فاعلية نموذج GPT-2 ذي الضّبط الدّقيق على غيره من النّماذج المستخدمة، وكانت نتائج نماذج GPT-3 الأقلّ من حيث الشّاعِرية. وفي حُدود عِلْم الباحثين، فإنّ هذه الدّراسة هي الأولى من بين أدبيات البحث التي وظّفت نماذج GPT-3 ذات الضّبط الدّقيق لتوليد الشِّعر العربي.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

INTERPRETING THE RELEVANCE OF READABILITY PREDICTION FEATURES

Safae Berrichi[1], Naoual Nassiri[2], Azzeddine Mazroui[1] and Abdelhak Lakhouaja[1]

(Received: 10-Nov.-2022, Revised: 3-Jan.-2023 and 31-Jan.-2023, Accepted: 13-Feb.-2023)

ABSTRACT

Text readability is one of the main research areas widely developed in several languages, but it is highly limited when dealing with the Arabic language. The main challenge in this area is to identify an optimal set of features that represent texts and allow us to evaluate their readability level. To address this challenge, we propose in this study various feature selection methods that can significantly retrieve the set of discriminating features representing Arabic texts. The second aim of this paper is to evaluate different sentence-embedding approaches (ArabicBert, AraBert and XLM-R) and compare their performances to those obtained using the selected linguistic features. We performed experiments with both SVM and Random Forest classifiers on two different corpora dedicated to learning Arabic as a foreign language (L2). The obtained results show that reducing the number of features improves the performance of the readability-prediction models by more than 25% and 16% for the two adopted corpora, respectively. In addition, the fine-tuned Arabic-BERT model performs better than the other sentence-embedding methods, but it provided less improvement than the feature-based models. Combining these methods with the most discriminating features produced the best performance.

KEYWORDS

Readability, Feature selection, Sentence embedding, Arabic language, Education.

1. INTRODUCTION

Arabic is one of the most used languages in the world. It is used by more than 400 million people[1] and is the official language of more than 20 countries. Arabic-language processing has attracted much more interest during this century. Many studies have focused on different aspects of Arabic-language processing, such as morphological analysis, resource construction, machine translation, sentiment analysis and readability assessment [1][2][3]. This last field of research (readability assessment), which is widely investigated in several other languages, represents an emerging field of research for Arabic.

Readability refers to the ease with which a reader can understand a written text. It can be assessed using a supervised learning approach, which is one of the most successful branches of Machine Learning (ML). This consists of training a predictive model using a set of data samples that are represented by a number of linguistic features [4]. These vectors are labeled with their classes, which correspond in the case of readability to the difficulty level of the text.

More recently, deep neural networks have been proposed to predict readability of texts in languages with large linguistic resources, such as English [5]. Although the model performance improves when the training corpora are enriched with additional data, this approach is not always adopted given that the readability annotation is time-consuming and has a high cost. An alternative approach to improve the prediction-model performance is to represent the text as embedded vectors using neural models. The latter learn semantics at the sequence level by considering all words in the document.

On the other hand, despite the advantages of ML-classification models that use a variety of linguistic features motivated by the language, they still suffer from several challenges [6]. Indeed, the use of relevant features to represent a text represents a great challenge [7]. Researchers are currently interested in developing solutions that extract the most discriminating features from the vectors that

[1] https://www.internetworldstats.com/stats7.htm (last visited on 09/10/2022)

1. S. Berrichi, A. Mazroui and A. Lakhouaja are with Department of Computer Science, Faculty of Sciences, Mohammed First Uni., Oujda, Morocco. Emails: berrichi.safae@gmail.com, azze.mazroui@gmail.com and abdel.lakh@gmail.com
2. N. Nassiri is with Engineering and Sustainable Development Team, Ibn Zohr Uni., Higher School of Technology, Dakhla. Morocco. Email: naoual.nassiri@gmail.com

represent the original texts. Feature selection is one of the methods used to overcome such a problem. This method consists in selecting the relevant features and eliminating the irrelevant or redundant ones [8]. Representing a text with the most discriminating features can improve the performance by increasing the generalization ability and the classification accuracy.

The contributions of this research can be summarized as follows. First, we identify subsets of the most relevant linguistic features used in Arabic text-readability measurement. Thus, we examine different feature-selection methods, individually and in combination, to determine their impacts on readability prediction models and to identify the optimal feature vector that achieves good performance. A second contribution aims to evaluate various sentence-embedding approaches, such as AabicBert, AraBERT and XLM-R. So, we first develop a readability-measurement model based entirely on one of these embedding approaches. Then, we develop a model that combines these embedded vector representations with different linguistic feature sets selected and judged relevant in this study.

To validate whether the obtained conclusions are independent of the used corpus, we evaluated our first experiments on two corpora. The first one, composed of 321 texts, was collected from the Aljazeera- Learning website[2] for learning Arabic and the second, consisting of 278 texts, was collected from the GLOSS platform[3] whose texts were developed by the Defense Language Institute Foreign Language Center[4] considered to be one of the top foreign language schools.

The remainder of this paper is structured as follows. Section 2 provides the general background related to readability-measurement techniques and feature-selection methods. Section 3 describes the basic concepts of feature selection and the most popular associated techniques adopted in this work. In the same section, the dataset used in this study, as well as the initial feature vector, are described. Section 4 presents the feature-selection results based on an ML approach, while deep-learning-based results are represented in Section 5. The last section is devoted to the conclusion and some thoughts on future work.

2. RELATED WORK

In the field of text-readability classification, many features have been used. However, few works have focused on identifying relevant features representing a document. In this section, we review recent work on measuring the readability of Arabic foreign language (L2) texts that focuses on linguistic features, as well as those that instead of linguistic features use the raw embedding of the input text. We also review studies related to techniques for selecting the most discriminating features.

2.1 Readability Assessment Early Approaches

In 2014, Forsyth described in his thesis [9] a system that consists of automatically predicting the readability of Modern Standard Arabic (MSA). The study is based on a corpus retrieved from the GLOSS platform and consisting of 179 texts. He incorporated lexical and morphological features in the model-generation process. In total, he generated a high-dimensional vector containing 162 features. Based on a cross-validation, he reported a maximum F-score value of 0.78 for three classes (easy, medium and difficult). In the same year, the authors of [10] described a study to evaluate whether a given text is suitable for an MSA learner as L2 using their own corpus. They focused on the vocabulary content of learners' programs and texts, as well as other word-related features. The model achieved an accuracy of about 60%. This study adopted a vector that was significantly less voluminous than Forsyth's (10 features).

Saddiki et al. conducted a study in 2015 [11] in which they evaluated the usefulness of lexical and morphological features for predicting readability. They gathered a corpus from the GLOSS platform consisting of 251 texts and compiled 35 low-complexity features in order to establish a baseline for future research on readability assessment. Their findings suggest that a small set of easily calculable features might be indicative of the reading level of a text. They reported a maximum accuracy of 73.31% and a maximum F-score of 0.73. Their F-score value is close to the value obtained by Forsyth (0.78) using only 35 features instead of the 165 features used by Forsyth.

[2] https://learning.aljazeera.net/en (last visited 09/10/2022)

[3] https://gloss.dliflc.edu/ (last visited 09/10/2022)

[4] https://www.dliflc.edu/ (last visited 09/10/2022)

The same authors conducted a study in 2018 [12] in which they used 146 features based on a GLOSS corpus composed of 576 MSA texts. Their best results reached 72.4% and 0.61 in terms of accuracy and F-score, respectively. These performances are lower than those obtained by the two previous works [9], [11]. In the same period, Nassiri et al. presented a study [13] in which they gathered a GLOSS corpus comprising 230 texts. They introduced a set of 170 features to represent a text. They reported an F-score of 0.9 when testing on the training data. This study suffers from a lack of generality, since the authors report results obtained by testing on the training data. Thus, these results are not comparable to the performances of the previously mentioned studies that reported evaluation results based on the use of conventional training and testing corpora practices. Finally, in 2021, Nassiri et al. presented a study [3] in order to identify a smaller set of features that could provide good readability-prediction accuracy. They eliminated features having low predictive weights, to end up with 76 relevant features and obtain an accuracy score of 86.15% on the test data.

Concerning the evaluation of readability in other fields besides education, we have some works on health texts' readability. In 2018, Al-Aqeel et al. [31] presented a study to assess the readability of written medicine information in terms of both vocabulary use and sentence structure. They assessed readability according to three difficulty levels (easy, intermediate and difficult) for 4,476 sentences. Looking to assess the quality and readability of the Arabic health information about COVID-19, in 2021, Halboub et al. [32] evaluated a set of websites. They concluded that practically all of the Arabic health information available on COVID-19 is not easily readable and understood by the general Arabic-speaking population. In 2022, Jasem et al. [33] conducted a study with the goal of evaluating Arab websites dedicated to breast cancer and recommended ways of improving engagement and access to health information. The results led them to conclude that, in general, the readability scores indicate that the websites are above the recommended reading level.

Most of these health-data dedicated studies are using traditional readability formulae (calculating reading scores) to evaluate the difficulty levels instead of machine-learning approaches and this is due to the limitation of unavailability of annotated data in this field to train supervised models.

2.2 Deep-learning-based Readability Assessment

In most state-of-the-art studies, researchers continue to opt for statistical ML techniques. These approaches are the most appropriate ones given the lack of large annotated corpora for readability and since large amounts of data are generally required to successfully use deep-learning architectures that use text embedding as input. As a result, studies based on these techniques are scarce for Arabic. In this sub-section, we will review some recently published works based on these new techniques.

In languages for which large amounts of data exist, readability-prediction techniques based on deep-learning models are emerging, unlike other languages such as Arabic, focusing on the English language, Deutsch et al. [5]. More recently, Lee et al. reported a study [15] based on the same concept of augmented deep-learning by combining linguistic features with transformers. They reported results supporting the hypothesis that the use of hand-crafted features improves the performance of deep-learning models on smaller datasets.

Concerning the Arabic language, the works are even scarcer. In 2021, Khallaf et al. [16] presented an approach to predict the difficulty of MSA sentences. They compared the performance of different types of sentence embedding (fastText, mBERT, XLM-R and Arabic-BERT) and compared them to traditional linguistic features, such as PoS tags, dependency trees, readability scores and frequency lists for language learners. Their best results were obtained using Arabic-BERT. They reported results with macro-averaged F-1 of about 0.80 and 0.75 for the Arabic-Bert and XLM-R readability classification, respectively.

2.3 Feature Selection

The complexity of natural languages calls for textual feature vectors with high dimensionality. The latter makes the classification process considerably difficult [17] and is therefore considered one of the main challenges in this field.

Feature selection has been addressed in many studies. The objective of these studies is to improve the performance of the models and to provide faster and less complex models. This is performed by selecting subsets of features from high-dimensionality feature sets. The authors in [18] examined the

effect of some feature-selection methods; namely, Information Gain [19], Correlation [20], SVM selection, Gini index and Chi-Square [21] and their combinations on the performance of the SVM classifier for sentiment analysis in dialectal Arabic. The results of this study show that the SVM classifier achieved the best performance when the SVM selection method was used. On the other hand, the SVM classifier performed better when the two methods "correlation and SVM selection" were combined consecutively.

Similarly, Elhassan et al. [22] studied the impact of using the Information Gain and the Chi-Square selection techniques on the performance of Arabic-text classification models based on the Naïve Bayes and the SVM classifiers. The tests performed showed that both feature-selection techniques improve the performance of the models and that the Information Gain technique produces the best results.

A comparative study between some feature selection methods adopted to categorize Arabic texts was reported by [23]. In addition, an adjustment was carried out to the feature-selection approaches by grouping the selected methods (Term Frequency Inverse Document Frequency "TF-IDF", Chi-Square and Information Gain). Furthermore, a new method was proposed to select the most appropriate features. This method is based on semantic fusion and multiple words (SF-MW) to construct the features. The combination of the adopted feature-selection methods yields better results than the individually selected methods. Thus, the proposed SF-MW feature-selection method is promising, because it reduces features and achieves a better classification accuracy.

Based on the review of many previous studies related to readability prediction of Arabic texts (sub-section 2.2), we notice that most of these studies have focused on the evaluation of different classifiers on MSA texts, using a determined set of features. The relevance of the features that compose the readability-prediction vectors in Arabic has been addressed, to our knowledge, only recently [3], [16].

Table 1 summarizes the set of Arabic readability-assessment studies that we have reviewed in this sub-section. The summary contains the distribution of the features categories (discussed in sub-section 3.3) in each study, the feature-selection method adopted (if it exists) and the used classifier(s). From these sets, we have chosen 70 features that we can implement based on the available tools and resources.

Table 1. Arabic readability-assessment studies.

Study	Features	RTF	MF	PDF	FLF	PBF	Selection method	Classifier
(Forsyth, 2014) [9]	162	✓	✓	✓	✓	✓	–	TiMBL [34]
(Cavalli-Sforza et al., 2014)[10]	10	✓	✓	✓			–	K-means
(Saddiki et al., 2015) [11]	35	✓	✓	✓	✓		–	Random Forest, SVM,...etc.
(Nassiri et al., 2017) [13]	170	✓	✓	✓	✓	✓	–	Random Forest, SVM,...etc.
(Saddiki et al., 2018) [12]	146	✓	✓	✓	✓	✓	–	Random Forest, SVM,...etc.
(Nassiri et al., 2021) [3]	76	✓	✓	✓	✓	✓	Correlation	Random Forest, SVM,...etc.
(Khallaf et al., 2021) [16]	5	✓		✓			Recursive Feature Elimination	Random Forest, SVM,...etc.

3. METHODS AND TOOLS

In this section, we will present the methods that we have adopted to address the problem of the suitability of the linguistic features used to assess the difficulty of a text and the impact of using sentence embedding with/without incorporating these features. We will also present the algorithms used to build the readability-prediction models and the used evaluation metrics. The data we have used in this study is also described in this section.

3.1 Relevant Feature-selection Methods

To overcome the problem of the usefulness of feature vectors used as input in a classification task, it is

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

usually recommended to use specific methods. This leads to enhanced training performance in some instances. These methods are usually divided into feature-extraction methods and feature-selection methods. The main distinction between these two categories is that feature-extraction methods combine the original features to generate new feature sets, while future-selection methods extract feature subsets from the original one. In this study, we will focus on the second category which is based on feature-selection methods. These methods are generally divided into three sub-categories: filter, embedded and wrapper methods. As illustrated in Figure 1, these methods select a subset of features in different manners.

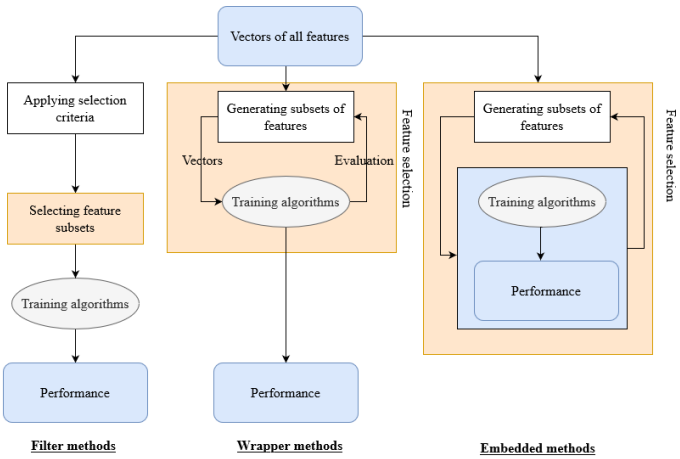


Figure 1. Process of feature-selection methods.

Given the wide range of existing feature-selection methods, choosing the best method for a given problem is a challenge. Furthermore, using these three different variations in feature selection allows us to take advantage of each algorithm that has a particular property and identify the method that captures the most discriminating features. In the following, we describe some reference methods that have shown successful performance and that we adopted in this study:

1) **Filter methods:** these methods are fast, scalable, computationally simple and independent of the classifier. They allow to ignore the dependencies between the feature selection and the classification algorithm. Among these methods, we cite:

   a) Information Gain (IG): a univariate filter that calculates the mutual information for each attribute and each class. Each feature will be ranked according to its information-gain value. Basically, the higher is the value, the more informative the feature is.
   b) Chi-square: a statistical technique used to determine whether there is a significant dependence between two categorical variables. The Chi-square test of independence attempts to determine which of the null hypothesis (independent variables) and the alternative hypothesis (dependent variables) is valid.

2) **Wrapper methods:** these methods require a training step to perform the feature selection. Although these methods are more expensive than filter methods due to their interactions with the classifier, they tend to perform better. The common wrapper method used in this study is based on recursive feature elimination for support vector machines (SVM-RFE) [24]. This method uses the weights of the SVM as a ranking criterion. The concept of SVM-RFE consists of training an SVM classifier in an iterative process and then exploiting the weights in the SVM solution to select some features and eliminate others.

3) **Embedded methods:** the concept of this last category is based on a combination of filter and wrapper methods. They use classification algorithms with an integrated feature-selection capability. They are less expensive in terms of computation than wrapper methods. Among these methods, the inherent features of the SVM (IF-SVM) and Random Forest (IF-RF) algorithms are adopted in this study.

3.2 Classification Algorithms

The Random Forest (RF) algorithm comprises a set of individual decision trees, each one trained on a

random subset of the training data. For a classification problem, the final prediction of RF is determined by a majority vote of the trees. By applying this method, RF reduces the problem of overfitting while maintaining good performance. In addition, it facilitates the interpretation of the results, since the features influencing the predictions can be identified and ranked according to their importance.

The SVM is a classification algorithm based on the statistical-learning theory [25]. It can be used for both linearly separable and non-linearly separable data. It uses a kernel function to project the input data into a high-dimensional space and subsequently determine a weight vector that represents the normal hyperplane for performing binary classification with minimal error rate. The kernels used in SVMs can be linear or non-linear. Non-linear kernels use the polynomial function or the radial basis function (RBF). To the best of our knowledge, the research community generally uses linear kernels to select features by SVMs.

Given the wide use of SVM and RF classifiers in Natural Language Processing (NLP) applications, especially in readability measurement, we adopted them to test the effectiveness of our proposed approaches. Similarly, since SVM feature selection relies on the linear kernel (IF-SVM), we chose to evaluate our models using the same kernel.

3.3 Data and Feature Description

In this study, we used MSA educational corpora intended for learning Arabic as L2. The texts of the latter are labeled with difficulty levels by experts using the ILR scale [26]. The three levels that we used can be interpreted as follows: easy (level 1 and level 1+), medium (level 2) and difficult (level 2+ and level 3). These texts are freely available in two different platforms, from which we have collected two corpora as follows:

1) GLOSS-Reading (GR): a corpus collected from the GLOSS platform which offers thousands of lessons, for independent learners, in dozens of languages, including Arabic. We collected 278 MSA texts annotated according to the three levels of difficulty described above. This corpus comprises a total of 4,666 sentences and 95,469 tokens.
2) Aljazeera-Learning (AL): Aljazeera website for learning the Arabic language also presents educational texts. We have collected from this site 321 texts annotated according to the three levels of difficulty. This last corpus contains 2,442 sentences and 49,345 tokens.

Tables 2 and 3 present in detail the statistics of these corpora according to the three levels of difficulty.

Table 2. Statistics of GLOSS-Reading corpus.

Level	**Texts**	**Sentences**	**Tokens**	**Average Sentence Length**
Easy	66	1,237	11,462	9.26
Medium	95	1,406	33,264	23.65
Difficult	117	2,023	50,770	25.09
Total	**278**	**4,666**	**95,469**	**20.46**

Table 3. Statistics of Aljazeera-Learning corpus.

Level	**Texts**	**Sentences**	**Tokens**	**Average Sentence Length**
Easy	232	1,277	20,840	16.31
Medium	54	378	7,646	20.22
Difficult	35	787	19,859	25.23
Total	**321**	**2,442**	**49,345**	**20.20**

When analyzing these two tables, we can notice that the used corpora suffer from an imbalance of data, especially for Aljazeera-Learning corpus. Regarding the average sentence length, calculated in terms of the number of sentences divided by the number of words, we conclude that it is a very important indicator, since it increases from one level to another for the two corpora.

Textual features associated with the degree of readability can be simple attributes, such as text length

or average word length; or more complex attributes, such as those related to grammatical categories.

The list of features that we used in this study is inspired from [3]. The authors of this work started with 170 features and found that some ratio features that are combinations of other existing features and some features that have the same value for all the texts are not discriminating for the readability prediction task; so they reduced the set from 170 to 76. So, we recompiled many of these features based on the PoS (Part of Speech) tags of the MADAMIRA analyzer [27]. In total, we have used a set of 70 features. This feature set was organized along two dimensions depending on the depth of processing required to extract them:

1) **Raw Text Features (RTF**): many formulae using raw-text features have been adopted and successfully adapted in English and other languages. Their popularity is due to their simplicity to compute and understand. In this category, we have chosen three features; the number of sentences, the number of words and the number of characters in a text.

2) **Features extracted after morphological analysis:** since readability is strongly influenced by vocabulary and word-level information, providing lexical and morpho-syntactic information at the word level can improve predictions. Features related to this different morpho-syntactic information are grouped into a set of sub-categories:

    a) *Morphological Features (MFs)*: these represent a sub-category composed of 5 features based on the distribution of vocabulary in a text. These five features are: the number of lemmas in the text, the number of stems, the number of frequent lemmas (lemmas that appear more than once in a text), the number of ambiguous lemmas (lemmas that have two different PoS tags in the same text) and the number of closed-class tokens[5].

    b) *PoS Dispersion Features (PDFs)*: these from a sub-category composed of 15 features extracted from PoS tags which examines the presence of different grammatical categories in the text (e.g. the number of verbs and the number of nouns).

    c) *Frequent Lemmas' Features (FLFs)*: this subcategory is composed of 10 features computed on the basis of the frequency dictionary (a dictionary containing the list of the 5,000 most frequent lemmas in the Arabic language with their morphological information); for example, the average dispersion of frequent lemmas or the average rank of frequent lemmas. For this sub-category, a lemma is considered frequent if it appears in the frequency dictionary.

    d) *PoS Based Frequency Features (PBFs)*: this is the largest sub-category in this study, comprising a total of 37 features. It represents the ratio of individual frequent grammatical categories (which appear in the frequency dictionary) to words in the text; for example, the average rank of frequent nouns and the maximum rank of frequent verbs.

3.4 Adopted Process

Figure 2 describes the approach implemented in this study. The process is performed in several steps. First, we represent each input text by a vector of 70 features. Given the importance of using the most discriminating features, we first applied the IG, Chi-square, IF-RF, IF-SVM and SVM-RFE feature-selection methods on the original vectors (based on the 70 first features). Then, we analyzed and compared the different feature sets obtained and subsequently we selected the most discriminating ones in the field of Arabic text-readability prediction. The models obtained using the five feature-selection methods and those obtained using the different combinations were evaluated based on the SVM and RF classifiers. Finally, we evaluated the impact of sentence embedding on the performance of readability predictions and we combined them with the best linguistic features obtained in the ML-based experiments.

3.5 Evaluation Measures

The performance of our models was evaluated based on the calculation of accuracy and F-score.

[5] a closed-class token is a token that belong to a grammatical category having a finite list such as prepositions

These measures are calculated based on the following formulae:

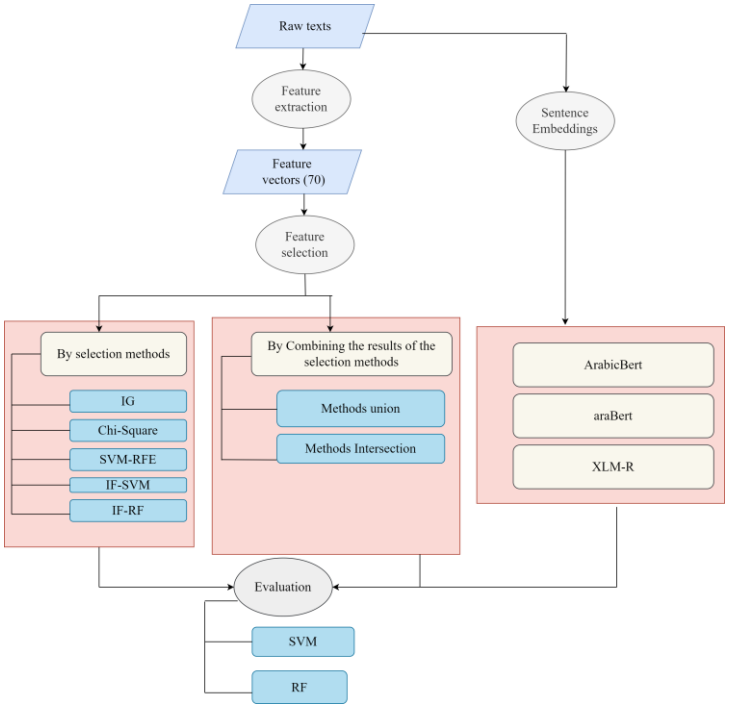


Figure 2. Adopted process for feature extraction.

$$Accuracy = \frac{number\ of\ correctly\ predicated\ documents}{total\ number\ of\ documents\ in\ the\ test\ set} * 100$$

$$Precision_i = \frac{number\ of\ documents\ correctly\ assigned\ to\ the\ class\ C_i}{total\ number\ of\ documents\ assigned\ to\ the\ class\ C_i}$$

$$Precision = \frac{\sum_{i=1}^{n} Precision_i}{n}$$

$$Recall_i = \frac{number\ of\ documents\ correctly\ assigned\ to\ the\ class\ C_i}{number\ of\ documents\ belonging\ to\ the\ class\ C_i}$$

$$Recall = \frac{\sum_{i=1}^{n} Recall_i}{n}$$

$$F - score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

where $Precision_i$ and $Recall_i$ represent, respectively, the precision and the recall of the class C_i and n represents the number of classes.

4. STATISTICAL ML-BASED EXPERIMENTS

4.1 Feature Selection Based on the AL Corpus

The goal of this paper is to select the most relevant features needed to predict the readability of an Arabic text. For this purpose, we conducted a set of experiments to examine the impact of the feature-selection methods discussed in Section 3 on the readability measurement of Arabic texts on the AL corpus. This methods rank the 70 features that we adopted in this study by their relevance degree. We compared the results of the selected features for each selection method with those obtained by the original "BaseLine" classification model (obtained using all the 70 features). Based on several experiments, we found that the best performances of all the selection methods are generally obtained with the 20 most informative features. The subset selected for each selection method is therefore made of the 20 most discriminating features. The parameters use in each selection method are those used by default in the scikit-learn library[6]. For RFE, we used the supervised learning SVM

[6] https://scikit-learn.org/stable/modules/feature_selection.html

estimator with kernel type and a 10-value regularization. To select the discriminative features by the RF machine-learning model that were trained and tested in both RF and SVM (IF-RF and IF-SVM), we kept the same parameters as those used in scikit-learn (the number of trees in the forest is 100 with a gini criterion and max_features to be taken into account when searching for the best split being sqrt). In the IG method, sklearn.feature_selection.mutual_info_classif was used with the default parameters and finally Chi-square sklearn.feature_selection.chi2 was used in the Chi-square method. All of these parameters are also retained in the classifiers and the remainder of the experiments to prevent the impact of the model parameters on the selection.

In the remainder of this experiment, we will evaluate the performance of each selected feature set. Thus, we adopted a random distribution of the AL corpus into 80% for training and the remaining 20% for testing. This distribution is adopted instead of 10-cross validation in order to evaluate the feature performances on the same data sets and this prevents the influence of data distribution on the results. In order to maintain the proportions of the three difficulty levels, we applied stratified sampling. The experiments were performed with both RF and SVM classifiers.

Table 4. Test results of the different feature sets on the AL corpus.

Model	**SVM Classifier**		**RF Classifier**	
	Accuracy	**F-score**	**Accuracy**	**F-score**
BaseLine	63.46%	63.54%	*86.53%*	79.17%
IG	*88.46%*	*83.21%*	*86.53%*	81.49%
Chi-square	73.07%	63.68%	84.61%	70.15%
SVM-RFE	**90.38 %**	**86.77%**	*86.53%*	**83.24%**
IF-RF	76.92%	72.51%	**88.46%**	*83.21%*
IF-SVM	84.61%	79.91%	86.53%	81.49%

Table 4 presents the accuracy and F-score of the "BaseLine" model and those of the models obtained using the most discriminating features selected by each of the five selection methods (IG, Chi-square, SVM-RFE, IF-RF and IF-SVM).

The results clearly show the relevance of using feature-selection methods in the readability prediction of Arabic texts. Indeed, except for the Chi-square method applied with the RF classifier, all the other methods perform better than the "BaseLine" model independently of the classifier used. Moreover, the selection method based on SVM-RFE reported the best results, since it improved the F-score by more than 23% for the SVM classifier and by more than 4% for the RF classifier. The IG and IF-RF methods reported the second best results using the SVM and the RF classifiers, respectively.

Concerning the embedded feature-selection methods IF-SVM and IF-RF, we notice that when these methods are embedded in their classification algorithm (IF-SVM with SVM and IF-RF with RF), the results are better compared to embedding them in another classification algorithm (e.g. IF-RF with SVM). We will exclude, in the rest of this paper, the Chi-square selection method that reported with RF classifier results lower than those reported by the "BaseLine" model.

The four subsets, each one comprising 20 features, generated by the four remaining feature-selection methods (IG, SVM-RFE, IF-RF and IF-SVM) are not identical. We thus analyzed these subsets to identify the most discriminating features. We initially find that only 39 of the 70 features used in the "BaseLine" model appear in these subsets. So, all the selection methods used in this study agreed that the remaining 31 features are irrelevant in the readability-prediction process. Moreover, only 24 features among the 39 are selected by at least two of the four selection methods. The complete list of these 24 features, in addition to three other features that we will discuss later, is presented in the appendix. We also note that all the five feature categories cited in sub-section 3.3 are present in the selected subsets. The dominant category is PBF with nine features, followed by PDF with eight features, as shown in the appendix.

On the other hand, we observe a strong presence of some features, since they appear in at least three of the four subsets. For this reason, we generated combinations of these feature subsets to determine the most informative ones. Table 5 presents the new subsets of features that result from these combinations.

Table 5. New feature subsets obtained on the AL corpus.

Sub-set	Reference	Features
Features that appear in at least one of the subsets	$F_{\cup 4methods}$	39
Features common to all four subsets	$F_{\cap 4methods}$	6
Features that appear in at least three subsets	$F_{\cap 3methods}$	12
Features that appear in at least two subsets	$F_{\cap 2methods}$	24
Union of the top five features from each subset	$F_{\cup 5best}$	12

The six features common to all methods, represented by the $F_{\cap 4methods}$ model, are illustrated in Table 6. Among the six common features, we have the number of lemmas which represents the size of the vocabulary and constitutes a feature strongly related to the difficulty of the text. Indeed, the more important is the vocabulary, the higher the difficulty of the text is. For the other features (numbers of nouns, properNoun and Noun/Open class token), we notice that much of the meaning of a text is in their nominal content.

Table 6. Common features between the four selection methods.

Features	Meaning
Lemma Count	The number of lemmas in a text
Noun Count	The number of nouns in a text
Adverb Count	The number of adverbs in a text
PropNoun Count	The number of proper nouns in a text
ClosedClassTokensCount	The number of closed class words in a text
Noun / Open Class Token	The ratio between nouns and the open class words in a text

After implementing these combinations between subsets of features (Table 5), we measured their impact on the readability prediction using the AL corpus. The performance of the models obtained using these subsets is compared with that of the initial "BaseLine" model, as well as with that of the SVM-RFE model, which is based on 20 features and produced the best results in the previous experiments. The test results are outlined in Table 7.

Table 7. Results of the different classification models on the AL corpus.

Model	SVM Classifier		Features	RF Classifier	
	Accuracy	F-score		Accuracy	F-score
BaseLine	63.46%	63.54%	70	*86.53%*	79.17%
SVM-RFE	90.38%	86.77%	20	86.53%	83.24%
$F_{\cup 4methods}$	71.15%	64.09%	39	**90.38%**	**86.77%**
$F_{\cap 4methods}$	88.46%	83.21%	6	84.61%	79.91%
$F_{\cap 3methods}$	90.38%	86.77%	12	86.53%	84.93%
$F_{\cap 2methods}$	82.69%	78.45%	24	86.53%	81.49%
$F_{\cup 5best}$	**92.30%**	**88.80%**	12	88.46%	84.93%

The results of this second experiment show that some combinations between the subsets obtained by the four feature-selection methods lead to better performance regardless of the classifier used. Indeed, the SVM classifier applied with the 12 most relevant features of the $F_{\cup 5best}$ subset outperforms the SVM- RFE model that uses 20 features by about 2% in terms of F-score. Thus, this leads us to suggest that these features are an indispensable benchmark for predicting readability (for details, see the feature set in the appendix). Similarly, for the RF classifier, the $F_{\cup 5best}$ model performed better than the SVM-RFE model. However, the best performance was recorded by the $F_{\cup 4methods}$ model, which combines all the features of the four selection methods.

4.2 Feature Selection Based on the GR Corpus

In order to determine whether the conclusions of the previous section remain valid regardless of the

corpus used, we examined the performance of the classification models on the GR corpus. The clustering of selected features generated the sets reported in Table 8.

From the selection results on this second corpus, we notice that 47 features from the initial 70 features were selected, unlike AL which only selected 39 features. The selection methods, based on the two corpora, agreed on a subset composed of 29 features. The 24 features that we mentioned in the appendix appear in this last subset obtained (composed of the 29 features).

Table 8. Number of features for the different combinations of the GR corpus.

Sub-set	**Reference**	**Features**
Features that appear in at least one of the subsets	$F_{\cup 4methods}$	47
Features common to all four subsets	$F_{\cap 4methods}$	3
Features that appear in at least three subsets	$F_{\cap 3methods}$	10
Features that appear in at least two subsets	$F_{\cap 2methods}$	20
Union of the top five features from each subset	$F_{\cup 5best}$	12

The performance of the readability-prediction models obtained using the new selected feature sets is reported in Table 9.

Table 9. Results of the different classification models on GR.

Model	**SVM Classifier**		**RF Classifier**	
	Accuracy	**F-score**	**Accuracy**	**F-score**
BaseLine	63.63%	63.27%	65.90%	66.27%
$F_{\cup 4method}$	52.27%	49.91%	79.54%	79.19%
$F_{\cap 4method}$	59.09%	59.49%	65.90%	63.99%
$F_{\cap 3method}$	50.0%	49.38%	72.72%	71.74%
$F_{\cap 2method}$	52.27%	51.49%	77.27%	76.58%
$F_{\cup 5best}$	**63.63%**	**65.91%**	**84.09%**	**83.11%**

Except for the $F_{\cap 4methods}$ model that contains only three features, the RF classifier applied to the other models provided better performance than the "BaseLine" model. Moreover, the $F_{\cup 5best}$ model achieves the best performance outperforming the "BaseLine" model by about 16% and this confirms the results reported on the AL corpus. Regarding the SVM classifier, we notice that the same model ($F_{\cup 5best}$) also improved the performance by about 2%, while the other selection models recorded a degradation compared to the "BaseLine" model.

Table 10 presents a list of the most discriminating features obtained by the $F_{\cup 5best}$ method, which provided the best performance. This list contains a total of 16 features with their meanings, which were identified by this method using the two corpora (AL/GR).

Table 10. List of features identified by $F_{\cup 5best}$ method using GR and AL corpora.

Feature	**Description**
Lemmas Count	Number of lemmas (without redundancy)
Closed Class Tokens Count	Number of tokens having a PoS tag belonging to a finite grammatical category (such as pronouns)
Nouns Count	Number of nouns
Tokens Count	Number of tokens
FreqLemmas Count	Number of frequent lemmas (lemma that appear more than once in the
AdjOpenClassTokens Ratio	The number of adjectives / the number of open class tokens (an open class token is a token that belongs to illimited grammatical category, such us nouns and verbs)
SL1	Number of sentences
Adverb Count	Number of adverbs

Stems Count	Number of stems
Preposition to Token	Number of prepositions / number of tokens
Chars Count	Number of characters
AMb1	Number of ambiguous lemmas (lemmas having two or more different PoS tags in the same text)
Noun Open Class	Number of nouns / number of open class tokens
Median Dispersion of Frequent Types	Frequent types are lemmas that appear in the frequency dictionary, having each one a dispersion value in the Arabic dictionary. This feature consists in calculating the means of all the retrieved values for all the lemmas composing a text.
Range of ranks of frequent adjectives	Frequent adjectives are ranked in the frequency dictionary; we calculate the range of all the ranks of a text adjectives.
The maximum rank of frequent nouns	For this feature, we get the ranks for all the frequent nouns (appearing in the frequency dictionary) and we get the maximum value. This means that we are getting the position of the most frequent noun in a text in the Arabic language represented by the frequency dictionary.

When analyzing the subsets of features obtained from the two corpora, we found differences in the selected features. This leads us to test a new subset C_U obtained by grouping the features identified by the two corpora. The objective of this new combination is to identify the essential features affecting readability regardless the nature of the corpus used. Thus, if we note F_{C1}/M_{C2}, the model using the C1 corpus to select the features and the C2 corpus to build the classification model (training and testing phases), we performed several experiments by choosing C1 and C2 among the AL and the GR corpora. We report in Table 11 the results of the best performing models only, hence the differences observed (last column) between the subsets of features adopted for the different methods.

Table 11. Results of the selected models by AL and GR.

SVM Classifier			
Experiment	**Accuracy**	**F-score**	**Model**
$\boldsymbol{F_{AL}/M_{AL}}$	**92.30%**	**88.80%**	$F_{\cup 5best}$ (12 features)
$\boldsymbol{F_{GR}/M_{AL}}$	88.46%	84.93%	$F_{\cap 4methods}$ (10 features)
$\boldsymbol{C_U/M_{AL}}$	90.38%	88.03%	$F_{\cap 3methods}$ (16 features)
$\boldsymbol{F_{AL}/M_{GR}}$	**68.18 %**	**69.72%**	$F_{\cap 4methods}$ (6 features)
$\boldsymbol{F_{GR}/M_{GR}}$	63.63%	65.91%	$F_{\cup 5best}$ (12 features)
$\boldsymbol{C_U/M_{GR}}$	63.63%	65.17%	$F_{\cap 4methods}$ (7 features)
RF Classifier			
Experiment	**Accuracy**	**F-score**	**Model**
$\boldsymbol{F_{AL}/M_{AL}}$	**90.38%**	**86.77%**	$F_{\cup 4methods}$ (39 features)
$\boldsymbol{F_{GR}/M_{AL}}$	**90.38%**	**86.77%**	$F_{\cap 4methods}$ (3 features)
$\boldsymbol{C_U/M_{AL}}$	88.46%	84.93%	$F_{\cap 3methods}$ (16 features)
$\boldsymbol{F_{AL}/M_{GR}}$	79.54 %	78.99%	$F_{\cup 4methods}$ (39 features)
$\boldsymbol{F_{GR}/M_{GR}}$	**84.09%**	**83.11%**	$F_{\cup 5best}$ (12 features)
$\boldsymbol{C_U/M_{GR}}$	79.54%	79.19%	$F_{\cup 5best}$ (12 features)

We observe that, for the SVM classifier, the best performance was reported by the F_{AL}/M_{AL} model built from the AL corpus and based on the features selected in this same corpus. Indeed, the substitution of these features by those selected for GR does not lead to an improvement of the performance (F_{GR}/M_{AL}), while the implementation of the features selected by AL on the GR corpus (F_{AL}/M_{GR}) outperforms the results provided by the F_{GR}/M_{GR} model by about 4% in terms of F-score. These results confirm the relevance of the features selected in the previous experiment based on the AL corpus. These observations do not hold for the RF classifier. The best model was obtained by applying the three features selected by GR and trained/evaluated on AL

($F_{GR/MAL}$). These features are included in the set of 39 features that gave the same results ($F_{AL/MAL}$). Finally, we notice that the fusion of the features selected by the two corpora did not yield any improvement whatever the corpus used. From these results, we can conclude that the best feature sets are those selected by the AL corpus and reported in the appendix.

5. DEEP-LEARNING-BASED EXPERIMENTS

In parallel with the use of linguistic characteristics to model readability, the representation of a text as an embedding vector using neural models represents another alternative proposed by researchers [16].

We select in this section a number of Arabic transformer models that generate sentence-embedding vectors:

1) AraBERT: is a pre-trained Bert Embeddings model available at "Hugging face transformers"[7]. AraBERT was trained using a combination of 70 million sentences from different resources (Arabic Wikipedia, Arabic corpus [28], …etc.). This model contains both MSA and Dialectal Arabic (DA). We have used the AraBERTv0.1-base model with the same parameters used in [30].
2) ArabicBERT: is another pre-trained BERT model[8] using a concatenation of a filtered subset from "Common Crawl" and a dump of Arabic Wikipedia totaling 8.2 billion words. Different pre-trained BERT models are used. We have selected in this study, the bert-base-Arabic model.
3) XLM-R [29]: is a multi-lingual version of BERT model, trained on "Common Crawl" data instead of Wikipedia with slightly different parameters. The model is trained on 100 different languages and contains approximately 2.5 TB sentences. The main parameters for learning cross-linguistic representations are those mentioned in [29].

These pre-trained models are used in this experiment to represent the text by sentence embedding. Each of them is compared with the models based on the linguistic features studied in the previous subsection. In addition, we investigated whether combining our linguistic feature set with deep-learning models could improve the performance of readability assessment. For this purpose, we have combined the most discriminating feature sets with different sentence-embedding models.

We first measured the performance of prediction models based on information-rich sentence embeddings as a separate feature set (AraBert, ArabicBert and XLM-R) and we compared them with the baseline model based on the 70 linguistic features.

To evaluate these models, we used the GR corpus according to the same distribution adopted in the previous experiments. The classification for the four models was performed with the RF classifier. Table 12 presents the accuracy and F-score results of these models.

Table 12. Results using the initial linguistic features and sentence embedding.

Model	Accuracy	F-score
BaseLine	65.90%	66.27%
AraBert	77.85%	78.34%
ArabicBert	**78.24%**	**79.36%**
XLM-R	71.60%	73.08%

All the three neural models trained using contextual sentence embedding outperformed the baseline model based only on hand-crafted linguistic features. Moreover, the ArabicBert model reported the best performance with an F-score about 13% higher than that of the baseline model. Based on these results, we hypothesize that the semantic and syntactic knowledge implicitly encoded in BERT embeddings can be considerably more informative than traditional linguistic features in predicting reading difficulty. Therefore, this is a likely alternative for low-resource languages that possess limited or no NLP tools, such as a parse tree extraction tool and a semantic parser. On the other hand,

[7] https://huggingface.co/aubmindlab/bert-base-arabert (last visited 09/10/2022)

[8] https://huggingface.co/asafaya/ (last visited 09/10/2022)

the best performance reached an accuracy of 78.24%, given the limitation of the texts used in the training phase, which is also accentuated when using artisanal linguistic features.

We have evaluated the impact of combining linguistic features with the information obtained by the neural models. We thus adopted a simple approach that consists in considering the numerical output of the neural model as a new feature that is incorporated into the linguistic features. We performed tests on the best combinations of the feature subsets identified in the experiments of the previous sections (F∩3methods, F∪4methods and F∪5best). The performances of the readability-prediction models relating to these new combinations are presented in Table 13.

Table 13. Combination of different sets of features and sentence-embedding models.

Model	AraBert		ArabicBert		XLM-R	
	Accuracy	F-score	Accuracy	F-score	Accuracy	F-score
CombinedBaseLine	82.63%	83.39%	81.60%	82.27%	76.00%	77.57%
$F_{\cap 3methods}$	**82.67%**	**83.49%**	**82.32%**	**83.30%**	**77.51%**	**78.96%**
$F_{\cup 5best}$	82.65%	83.28%	81.94%	82.80%	76.75%	78.08%
$F_{\cup 4methods}$	81.17%	82.36%	82.67%	83.18%	75.64%	77.08%

We point out that in general, the combination of linguistic features with the numerical output of the neural model improves the performance of classification based only on linguistic features. Indeed, by comparing the results of Table 9 and Table 13, we find that, in terms of F-score, the performance of the classification based on the combination of linguistic features with the output numerical of one of the two neural models AraBert and ArabicBert has improved for all models.

Similarly, the results of Table 12 and Table 13 suggest that these combined models perform better than those using only the outputs of the neural models. The best performances are obtained by combining the numerical output of the AraBert model with the subset of linguistic features F∩*3methods*.

In order to analyze the results in more detail, we computed the confusion matrix of the best classification model that combines F∩*3methods* with AraBert. It is clear from Table 14 that there is a separation between the prediction levels (easy, intermediate and difficult). Indeed, all the errors are located between the neighboring levels.

Table 14. Confusion matrix of $F_{\cap 3methods}$ model incorporated with AraBert.

	Easy	Intermediate	Difficult
Easy	6	1	0
Intermediate	0	7	2
Difficult	0	1	10

Finally, we conclude that the F∩*3methods* model is composed of the most informative features compared to the other models, which in turn provides an indispensable benchmark for predicting readability.

6. CONCLUSION

In this study, we presented a set of experiments related to the selection of the most informative features for Arabic L2-readability measurement. The obtained test results showed that feature selection is a useful pre-processing tool that not only reduces the number of input features, but also increases the performance of prediction models.

We considered the three main types of selection methods (filter, wrapper and embedded) commonly used to retrieve the most discriminating features. We have thus applied these methods to the two corpora AL and GLOSS. Next, we examined the impact of each selection method on the performance of readability-prediction model and we compared it with the performance of the baseline model using 70 features.

We also demonstrated the relevance of the combination of features selected by the different methods; namely, "F∩*4methods*", "F∩*3methods*", "F∩*2methods*", "F∩*5best*" and "F∪*4methods*". These combinations

were also evaluated on the grouping of the features selected by the two corpora. The performance of all the experiments was evaluated using the two RF and SVM classifiers.

The use of sentence embedding is another approach used to represent a text. The last experiment of this work aimed to investigate the influence of these methods in the field of readability assessment. Thus, we have evaluated the incorporation of the best features that we have selected with these representations. In the case of limited corpus size, we have shown that the best results are obtained by incorporating both types of representation (sentence embedding and hand-crafted linguistic features).

Concerning our future directions of investigations, we are considering the evaluation of readability based on attested/evaluated reading comprehension of human readers. This is due to the fact that the actually available readability corpora are labeled based on experts' opinion, but not on reading comprehension tests. Gathering a large amount of training data, to enhance the performance of sentence embedding-based models and adopt deep-learning classifiers is also one of our future work paths. The readability of Arabic as a native language (L1) is also an emerging field of research; so it will be interesting to examine the impact of feature-selection methods on L1 and compare the results with those on L2, to check whether the L1 and L2 reading learners share common discriminating readability features.

REFERENCES

[1] M. Al-Ayyoub, A. A. Khamaiseh, Y. Jararweh and M. N. Al-Kabi, "A Comprehensive Survey of Arabic Sentiment Analysis," Inf. Processing and Management, vol. 56, no. 2, pp. 320–342, 2019.

[2] S. Berrichi and A. Mazroui, "Addressing Limited Vocabulary and Long Sentences Constraints in English–Arabic Neural Machine Translation," Arabian J. for Science and Engineering, vol. 46, pp. 8245–8259, 2021.

[3] N. Nassiri, A. Lakhouaja and V. Cavalli-Sforza, "Arabic L2 Readability Assessment: Dimensionality Reduction Study," J. of King Saud Uni., Comp. and Inf. Sci., vol. 34, pp. 3789–3799, 2022.

[4] V. Cavalli-Sforza, H. Saddiki and N. Nassiri, "Arabic Readability Research: Current State and Future Directions," Procedia Computer Science, vol. 142, pp. 38– 49, 2018.

[5] T. Deutsch, M. Jasbi and S. Shieber, "Linguistic Features for Readability Assessment," Proc. of the 15th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 1–17, Association for Computational Linguistics, Seattle, USA, 2020.

[6] D. D. Lewis, "Challenges in Machine Learning for Text Classification," Proc. of the 9th Annual Conference on Computational Learning Theory, pp. 1–ff, 1996.

[7] X.-D. Wang, R.-C. Chen, F. Yan, Z.-Q. Zeng and C.-Q. Hong, "Fast Adaptive K-Means Subspace Clustering for High-dimensional Data," IEEE Access, vol. 7, pp. 42639 – 42651, 2019.

[8] I. Guyon, S. Gunn, M. Nikravesh and L. A. Zadeh, Feature Extraction: Foundations and Applications, ISBN: 978-3540354871, Springer, 2008.

[9] J. N. Forsyth, Automatic Readability Prediction for Modern Standard Arabic, Ph.D. Thesis, Department of Linguistics and English Language, Brigham Young University, USA, 2014.

[10] V. Cavalli-Sforza, M. El Mezouar and H. Saddiki, "Matching an Arabic Text to a Learners' Curriculum," Proc. of the 5th Int. Conf. on Arabic Lang. Process. (CITALA), p. 10, Morocco, 2014.

[11] H. Saddiki, K. Bouzoubaa and V. Cavalli-Sforza, "Text Readability for Arabic as a Foreign Language," Proc. of the 2015 IEEE/ACS 12th Int. Conf. of Computer Systems and Applications (AICCSA), pp. 1–8, Marrakech, Morocco, 2015.

[12] H. Saddiki, N. Habash, V. Cavalli-Sforza and M. Al-Khalil, "Feature Optimization for Predicting Readability of Arabic l1 and l2," Proc. of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, pp. 20–29, DOI: 10.18653/v1/W18-3703, Melbourne, Australia, 2018.

[13] N. Nassiri, A. Lakhouaja and V. Cavalli-Sforza, "Modern Standard Arabic Readability Prediction," Proc. of the Int. Conf. on Arabic Language Processing (ICALP 2017), Part of the Communications in Computer and Information Science Book Series, vol. 782, pp. 120–133, 2018.

[14] N. Nassiri, A. Lakhouaja and V. Cavalli-Sforza, "Arabic Readability Assessment for Foreign Language Learners," Proc. of the Int. Conf. on Applications of Natural Language to Information Systems (NLDB 2018), vol. 10859, pp. 480–488, 2018.

[15] B. W. Lee, Y. S. Jang and J. H.-J. Lee, "Pushing on Text Readability Assessment: A Transformer Meets Handcrafted Linguistic Features," Proc. of the 2021 Conf. on Empirical Methods in Natural Lang. Process., pp. 10669–10686, DOI: 10.18653/v1/2021.emnlp-main.834, Dominican Rep., 2021.

[16] N. Khallaf and S. Sharoff, "Automatic Difficulty Classification of Arabic Sentences," Proc. of the 6th Arabic Natural Language Processing Workshop, Association for Computational Linguistics (WANLP 2021), pp. 105–114, DOI: 10.48550/arXiv.2103.04386, 2021.

[17] V. N. Vapnik, "Controlling the Generalization Ability of Learning Processes," Chapter 4 in Book: The Nature of Statistical Learning Theory, pp. 89–118, Springer, 2000.
[18] O. Al-Harbi, "A Comparative Study of Feature Selection Methods for Dialectal Arabic Sentiment Classification Using Support Vector Machine," Int. J. of Computer Science and Network Security, vol. 19, no. 1, pp. 167-176, January 2019.
[19] H. U˘guz, "A Two-stage Feature Selection Method for Text Categorization by Using Information Gain, Principal Component Analysis and Genetic Algorithm," Knowledge-based Systems, vol. 24, no. 7, pp. 1024-1032, 2011.
[20] M. A. Hall, Correlation-based Feature Selection for Machine Learning, Ph.D. Thesis, Department of Computer Science, University of Waikato, New Zealand, 1999.
[21] S. Bahassine, A. Madani, M. Al-Sarem and M. Kissi, "Feature Selection Using an Improved Chi-square for Arabic Text Classification," J. of King Saud Uni.-Comp. and Inf. Sci., vol. 32, no. 2, pp. 225–231, 2020.
[22] R. Elhassan and M. Ali, "The Impact of Feature Selection Methods for Classifying Arabic Texts," Proc. of the 2nd Int. Conf. on Comp. App. Inf. Secur. (ICCAIS), pp. 1–6, Riyadh, KSA, 2019.
[23] A. Elnahas, N. Elfishawy, M. Nour and M. Tolba, "Machine Learning and Feature Selection Approaches for Categorizing Arabic Text: Analysis, Comparison and Proposal," The Egyptian Journal of Language Engineering, vol. 7, no. 2, pp. 1–19, 2020.
[24] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, "Gene Selection for Cancer Classification Using Support Vector Machines," Machine Learning, vol. 46, no. 1, pp. 389–422, 2002.
[25] B. E. Boser, I. M. Guyon and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," Proc. of the 5th Annual Workshop on Computational Learning Theory, pp. 144–152, DOI: 10.1145/130385.130401, 1992.
[26] J. L. D. Clark and R. T. Clifford, "The FSI/ILR/ACTFL Proficiency Scales and Testing Techniques: Development, Current Status and Needed Research," Studies in Second Language Acquisition, vol. 10, no. 2, pp. 129–147, 1988.
[27] A. Pasha, M. Al-Badrashiny, M. T. Diab et al., "MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic," Proc. of the 9th Int. Conf. on Language Resources and Evaluation (LREC'14), vol. 14, pp. 1094–1101, Reykjavik, Iceland, 2014.
[28] I. A. El-Khair, "1.5 Billion Words Arabic Corpus," CoRR abs/1611.04033, arXiv: 1611.04033, DOI: 10.48550/arXiv.1611.04033, 2016.
[29] A. Conneau, K. Khandelwal, N. Goyal et al., "Unsupervised Cross-lingual Representation Learning at Scale," Proc. of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451, DOI: 10.18653/v1/2020.acl-main.747, 2020.
[30] A. Safaya, M. Abdullatif and D. Yuret, "KUISAIL at SemEval-2020 Task 12: BERT-CNN for Offensive Speech Identification in Social Media," Proc. of the 14th Workshop on Semantic Evaluation, pp. 2054–2059, DOI: 10.18653/v1/2020.semeval-1.271, Barcelona, Spain, 2020.
[31] S. Al-Aqeel, N. Abanmy, A. Aldayel, H. Al-Khalifa, M. Al-yahya and M. Diab, "Readability of Written Medicine Information Materials in Arabic Language: Expert and Consumer Evaluation," BMC Health Services Research, vol. 18, DOI: 10.1186/s12913- 018-2944-x, 2018.
[32] E. Halboub, M. S. Al-Ak'hali, H. M. Al-Mekhlafi and M. N. Alhajj, "Quality and Readability of Web-based Arabic Health Information on COVID-19: An Infodemiological Study," BMC Public Health, vol. 21, no. 1, pp. 1–7, 2021.
[33] Z. Jasem, Z. AlMeraj and D. Alhuwail, "Evaluating Breast Cancer Websites Targeting Arabic Speakers: Empirical Investigation of Popularity, Availability, Accessibility, Readability and Quality," BMC Medical Informatics and Decision Making, vol. 22, no. 1, pp. 1–15, 2022.
[34] W. Daelemans, J. Zavrel, K. van der Sloot and A. van den Bosch, "TiMBL: Tilburg Memory Based Learner," Technical Report, Version 6.3, ILK Research Group Technical Report.

APPENDIX: LIST OF FEATURES

Table 15. Common features between IG, IF-RF, IF-SVM, SVM-RFE and $F_{\cup 5best}$.

Category	Features	IF-RF	IF-SVM	IG	SVM-RFE	$F_{\cup 5best}$
RTF	Characters Count	✓		✓		
	Token Count	✓		✓	✓	✓
	Sentences Count		✓	✓		
MF	Stem Count	✓		✓		✓
	Lemma Count	✓	✓	✓	✓	✓
	Ambiguous Lemma Count	✓	✓	✓		
PDF	Noun Count	✓	✓	✓	✓	✓
	Verb Count	✓		✓		
	Adverb Count	✓	✓	✓	✓	✓
	Proper Nouns Count	✓	✓	✓	✓	
	Adjectives Count	✓	✓	✓		

	Closed-Class Tokens Count	✓	✓	✓	✓	✓
	Noun / Open Class Token	✓	✓	✓	✓	
	Adjectives to Open Class Token Ratio		✓	✓	✓	✓
	Preposition to Token Ratio				✓	✓
FLF	Frequent Lemmas Count	✓		✓	✓	✓
PBF	Ratio of frequent subordinating conjunctions to total subordinating conjunctions		✓		✓	
	Ratio of frequent prepositions to total prepositions		✓	✓	✓	
	Ratio of frequent demonstrative pronouns to total demonstrative pronouns			✓	✓	
	Maximum rank of frequent conjunctions		✓		✓	
	Maximum rank of frequent subordinating conjunctions		✓		✓	
	Average rank of frequent subordinating conjunctions		✓	✓		
	Median rank of frequent particles		✓		✓	
	Minimum rank of frequent proper names		✓	✓		
	Range of ranks of frequent adjectives		✓		✓	✓
	The maximum rank of frequent nouns					✓

ملخص البحث:

إنّ مقروئيّة النّصوص تمثّل أحد مجالات البحث التي تطوّرت على نطاقٍ واسع فيما يتعلّق بالعديد من اللُّغات، غير أنها محدودة الى حدّ كبير عندما يتعلّق الأمر باللّغة العربية. ويكمن التّحدي الرئيسي في هذا المجال في تحديد مجموعة مثالية من السِّمات التي تمثّل النّصوص وتسمح لنا بتقييم مُستوى مقروئيّتها. وللتّعامل مع هذا التّحدي، نقترح في هذه الدّراسة طرقاً متنوعة لانتقاء السِّمات تُمكّننا من استرجاع مجموعة السِّمات المميّزة للنّصوص باللّغة العربية. ويتمثّل الهدف الثاني من هذه الدراسة في تقييم طُرق مختلفة لتّضمين الجُمل ومقارنة أداءاتها بتلك التي يتمّ الحصول عليها من السِّمات اللُّغوية المُنتقاة. فقد أجرينا تجارب باستخدام مُصنِّفات (SVM) و (RF) على اثنتين من مجموعات البيانات المخصصة لتعلُّم اللُّغة العربية كلُغة ثانية (L2).

وتكشف النّتائج التي تمّ الحصول عليها أنّ تقليل عدد السِّمات يعمل على تحسين أداء نماذج توقُّع المقروئيّة بنسبةٍ تتجاوز 25% و 16% لمجموعتي البيانات المستخدمتيْن، على الترتيب. بالإضافة الى ذلك، يعمل نموذج (BERT) المزوَّد بإمكانية الضّبط الدّقيق والمخصَّص للُّغة العربية بأداءٍ أفضل مقارنةً بالنّماذج الأخرى المستخدَمة لتضمين الجُمل، لكنّه أدّى الى تحسُّن أقلّ عند مقارنته بالنّماذج القائمة على السِّمات. وقد أدّى الجَمع بين هذه الطّرق والسِّمات الأكثر تمييزاً للنّصوص الى الحصول على أفضل النّتائج.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

A DEEP DECISION FORESTS MODEL FOR HATE SPEECH DETECTION

Kennedy Malanga Ndenga

(Received: 2-Nov.-2022, Revised: 4-Feb.-2023, Accepted: 14-Feb.-2023)

ABSTRACT

Detecting and controlling propagation of hate speech over social-media platforms is a challenge. This problem is exacerbated by extremely fast flow, readily available audience and relative permanence of information on social media. The objective of this research is to propose a model that could be used to detect political hate speech that is propagated through social-media platforms in Kenya. Using Twitter textual data and Keras TensorFlow Decision Forests (TF-DF), three models were developed; i.e., Gradient Boosted Trees with Universal Sentence Embedding (USE), Gradient Boosted Trees and Random Forest, respectively. The Gradient Boosted Trees with USE model exhibited a superior performance with an accuracy of 98.86%, a recall of 0.9587, a precision of 0.9831 and an AUC of 0.9984. Therefore, this model can be utilized for detecting hate speech on social media platforms.

KEYWORDS

Hate speech detection, TensorFlow decision forests, Gradient boosted trees, Universal sentence embedding, National cohesion and integration commission.

1. INTRODUCTION

Social media has permeated our lives such that a majority of unstructured communications happen through these platforms. Social media provides a ready audience anytime. As a result, a large fraction of discourses that were initially verbally articulated have shifted to social-media platforms. Control of communication over social media is difficult to achieve or enforce, considering that the operators of social-media platforms are foreign companies operating from different legislative environments compared to those of their users. This problem is exacerbated by the extremely fast flow and dynamism of information on social media. Therefore, sensitive and potentially harmful content can propagate through social-media platforms quickly without being detected. Hate speech is an example of such communication that can be difficult to deal with particularly when it is propagated through social-media platforms. Actually, Mr. António Guterres –the United Nations (UN) Secretary-General– describes Social media as a global megaphone for hate [1].

According to the UN's Strategy and Plan of Action on Hate Speech, hate speech is any kind of communication in speech, writing or behaviour that attacks or uses a pejorative or discriminatory language with reference to a person or a group on the basis of who they are. In other words, this happens based on their religion, ethnicity, nationality, race, colour, descent, gender or any other identity factor [1]. Hate speech has been identified as a trigger of violence and suffering in several parts of the world, including the Tigray region of Ethiopia, Guinea, Sri Lanka, …etc. [2]. In Kenya, political hate speech was blamed for the 2007/2008 post-election violence [3]. During the COVID-19 pandemic, social-media platforms contributed immensely to Sino-phobic hate sentiments, where an Asian community was blamed for the pandemic [34]. Khan et al. add that social-media platforms accelerated propaganda related to the Shaheen Bagh protests in New Delhi against the National Register of Citizens, Citizenship Amendment Act and National Population Register [34].

Propagation of hate speech over social-media platform is relatively a new phenomenon, considering that social-media platforms are recent disruptive technologies. Hate speech can be propagated on social media through: text messages, pictures, videos, emoji or emoticons. Sometimes, hate speech could be obfuscated in online content that seems ordinary. Various approaches, including intentional misspelling by swapping characters, elongating words using many repeated letters or putting spaces between letters, have been used to obfuscate hate speech in social-media discourses [35]. Hiding of hate-text messages in images and hate sarcasm using images or videos are other techniques of hiding hate speech over social-media platforms [6].

K. Malanga Ndenga is with the Department of Pure & Applied Sciences, Kirinyaga Univ., Kutus, Kenya. Email: kmalanga@kyu.ac.ke

Poletto et al. [4] argue that explicitly defining hate speech is challenging because of widespread vagueness in the use of related terms, such as abusive, toxic, dangerous, offensive or aggressive language, that often overlap and are prone to strongly subjective interpretations. However, they go ahead and conclusively define hate speech as a content defined by its action; i.e., generally spreading hatred or inciting violence or threatening people's freedom, dignity and safety by any means and by its target – which must be a protected group or an individual targeted for belonging to such a group and not for his/her individual characteristics. Figure 1 depicts this definition.

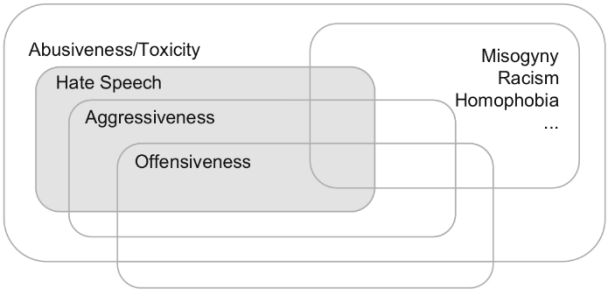


Figure 1. Relationships between hate speech and related concepts [4].

It is challenging to deal with hate speech that is propagated over social-media platforms as compared to that which is propagated over traditional media, like newspapers, magazines, TV, radio or billboards. Unlike in traditional media, online hate speech can be produced and distributed easily, at low cost and anonymously while having the potential to reach a global and diverse audience in real time [1]. The relative permanence of online content is also problematic when hateful discourse can resurface and regain popularity over time [1]. The UN concludes that efforts of understanding and monitoring the dynamics of hate speech across diverse online communities and platforms often stall given the sheer scale and diversity of the phenomenon, current technological limitations of automated monitoring systems and the opacity of online companies [5]. Therefore, online hate speech detection and control represent a phenomenon that deserves research attention.

The purpose of this research is to propose a model that could be used to detect political hate speech that is propagated through social-media platforms in Kenya. Such a model could be used as a tool of decision support for monitoring future social-media discourses. The study described in this work treats only the textual form of hate speech. The remaining parts of the article are organized as follows: Section two is a discussion of previous work that is related to this study, where various natural-language processing (NLP) techniques for hate speech detection are assessed; Section three describes the approach proposed by this research; Section four outlines the method of research experiments; Section five presents results from the experiments; Section six is a discussion of the results, while section seven concludes the study. This study has made three main contributions to knowledge. The first contribution is methodological, where a new method of hate speech detection has been described by the study. Proposing a new model for hate speech detection is the second contribution. Lastly, the study has introduced the Kenyan culture in the area of hate speech detection.

2. RELATED WORK

Hate speech detection has attracted the attention of researchers in the recent few years. Gomez et al. [6] studied hate speech detection in multi-modal publications formed by text and image data from Twitter. They found out that multi-modal models cannot outperform text models in detecting hate speech. Based on this finding, this study developed a hate-speech detection model using the text component of tweets from Kenyan political discourses.

Mullah et al. [7] reviewed machine-learning algorithms and techniques for hate speech detection in social media. They found out that the majority of existing research work on hate speech detection used classical machine-learning techniques as compared to ensemble and deep-learning techniques. They also identified some open challenges in hate speech detection which include: cultural variations, pandemic or natural disasters, data sparsity, imbalanced datasets and dataset availability concerns. They emphasized the need to take the campaign of hate speech prevention to other non-western parts of the world, since culture and tradition play a significant role in hate speech detection efforts [7]. Mullah et

al.'s [7] emphasis is agreeable, since contextual meanings of words in discourses vary from one cultural region to another. Thus, models trained on text datasets in one culture may have some degree of bias if used to test data from a different culture. Therefore, by experimenting with data from the Kenyan culture, models in this field are enriched.

Ombui et al. [3] studied the identification of hate speech in code-switched text messages by exploring the performance of different features across various machine-learning algorithms. They established that character level Term Frequency-Inverse Document Frequency performed best given a code-switched dataset of 25k annotated tweets using support vector machine algorithm as compared to six other conventional and two deep-learning algorithms.

Khan et al. [34] presented HcovBi-Caps –a Convolutional, BiGRU and Capsule network-based deep-learning model– to classify hate speech and evaluated it over two Twitter-based benchmark datasets; i.e., DS1 which was balanced and DS2 which was unbalanced. They found out that HCovBi-Caps showed comparatively better performance over the unbalanced dataset with precision, recall and f-score values of 0.90, 0.80 and 0.84, respectively. In another study, Khan et al. [36] introduced BiCHAT, a BiLSTM deep-learning model for hate speech detection. The model was trained and evaluated over three benchmark datasets; i.e., HD1, HD2 and HD3 extracted from Twitter. They found out that their BiCHAT model outperformed three state-of-the-art models used in studies with an improvement of 8%, 7% and 8% in terms of precision, recall and f-score, respectively. They acknowledge that the performance of their models in the two studies were based on evaluations done on non-diverse datasets in a non-multilingual set-up. Like Gomez et al. [6], Koutlis et al. [37] presented MemeTector, a multi-modal model for classifying images as memes or regular images. They proposed that their model could be utilized in online social environments to detect hate speech and disinformation. Aggarwal et al. [38] too proposed an approach to solve the problem of identifying hate memes. However, the two studies did not compare how their multi-modal models performed relative to text-only models in detecting hate speech [37]-[38].

Poletto et al. [4] systematically analyzed resources made available for hate speech detection models from the perspective of: their development methodology, topical focus, language coverage, among other factors. Their survey found out that datasets are available in several languages for hate speech detection, but focus on different topics. Like Mullah et al. [7], Poletto et al. [4] added that there is a challenge of developing hate speech detection architectures which are stable and well performing across different languages and abusive domains. They also noted that biases in the design and annotation of the training dataset, as well as topic biases in the training datasets as compared to the test and volatile nature of topics are factors to keenly consider when developing resources for hate speech detection.

Badjatiya et al. [8] investigated the application of deep neural network architectures for the task of hate speech detection and found out that embeddings learned from deep neural-network models when combined with gradient-boosted decision trees led to the best accuracy values. Dorris et al. [9] introduced the HateDefender –a hate speech and offensive language defence system– which consists of a detection model based on deep Long Short-term Memory (LSTM) neural networks that according to them can effectively detect hate speech with an average accuracy of 90.82%. Like Badjatiya et al. [8] and Dorris et al. [9], deep-learning techniques have also been used to develop models for detecting hate speech in this research. However, the approach in this study differs from those of the discussed studies, since it utilizes different embeddings used as discussed in Section 3.

Other works use the more recent and well-known type of DNN; namely, transformers, in particular the BERT model proposed by Google [22]. These techniques of language modeling are presented as a general model used for a variety of NLP tasks; for example, translation, question and response tasks. However, when used in transfer learning based on a pre-trained model, the model is fine-tuned using a special dataset more suitable for the task. For instance, Mozafari et al. [23] proposed a novel transfer learning approach based on BERT – an existing pre-trained language model – in order to overcome the problem of lack of sufficient amount of labelled hate speech data. Velankar et al. [24] presented baseline classification results using deep-learning models based on CNN, LSTM and Transformers. Using a multi-lingual BERT version fine-tuned with an annotated Marathi language dataset, Velankar et al. [24] showed that transformers outperformed other methods. Most recent works tackle hate speech detection using neural networks to train models of detection. This means that they usually need huge datasets so that their models can converge therefore making them not suitable with a relatively small dataset. It is

believed that models based on decision forests as applied in this study could overcome this limitation.

2.1 Techniques for Hate Speech Detection

Hate speech is a classification problem. Classical, ensemble and deep-learning classification techniques have previously been used to achieve hate speech detection with varying degrees of success [7]. Classical machine-learning algorithms require more structured data and are more dependent on human intervention to learn, whereby human experts determine the hierarchy of features through data labelling for the machine to understand differences between data inputs [39]. Examples of such algorithms include support vector machines (SVMs), Naive Bayes (NB), Logistic Regress (LR), Decision Trees (DTs) and K-Nearest Neighbour (KNN) [7]. Unlike classical machine learning, deep-learning algorithms automate much of the feature-extraction process, thus eliminating some of the manual human intervention required [39]. Deep-learning algorithms differ from machine-learning algorithms, since they require large datasets to learn reasonably, while machine learning requires less to learn [7]. On the other hand, ensemble learning schemes combine multiple base machine learning algorithms of any type – e.g. decision tree, neural network, linear regression model, …etc. – to make a decision, typically in supervised machine-learning tasks [40]. Classical machine-learning techniques are unable to effectively analyze some text datasets that are very large and not linearly separable; however, deep-learning algorithms have capabilities of effectively analyzing such datasets [7]. Since this study is dealing with unstructured text dataset, it was found prudent to apply deep-learning techniques to analyze the data.

Variants of deep learning that have been used for hate speech detection are mostly Deep Neural Network techniques, which include Convolution Neural Network, Recurrent Neural Networks, i.e., Long Short Term Memory (LSTM) and Gated Recurrent Units (GRUs) [7]. Deep neural networks have multiple hidden layers [41]. Long Short Term Memory (LSTM) networks [10] and their variants have been used by Gomez et al. [6], Dorris et al. [9], Ong [11] and Badjatiya et al. [8] for the classification of hate speech texts from social media. Long Short Term Memory networks are a special kind of Recurrent Neural Networks (RNNs) capable of learning long-term dependencies in a sequence (sentence) [12]. An RNN is a network with loops allowing information to persist and can be thought of as multiple copies of the same network, each passing a message to a successor [12]. Gomez et al. [6] used LSTM, because they believed that these algorithms provide a strong representation of tweet text data. Their LSTM models gave an f-score of 0.703, an area under the ROC of 0.732 and a mean accuracy of 68.4 for tweet text input. Ong [11] found out that Bidirectional LSTM (BiLSTM) convolutional neural network model was optimal with the highest f1-score.

Badjatiya et al. [8] performed experiments on a benchmark dataset of 16K annotated tweets and showed that three deep learning architectures i.e., FastText, Convolutional Neural Networks (CNNs) and Long Short Term Memory Networks (LSTMs) deep-learning methods, outperformed char/word n-gram methods by approximately 18 f1 points. On the other hand, Ombui et al. [3] explored both conventional and deep-learning classifiers and found out that Support Vector Machine algorithm yielded the best classification accuracy when classifying code-switched text messages. Unlike the previously discussed techniques, Keras and TensorFlow Decision Forests (TF-DF) were used to develop three models for this study; i.e., the Gradient Boosted Trees with Universal Sentence Embedding model, the Gradient Boosted Trees model and the Random Forest model.

3. PROPOSED APPROACH

For model development, Keras TensorFlow Decision Forests (TF-DF) were used to train neural network models for classifying tweeter text data. Keras is a deep-learning framework that makes it easier to run new experiments [13]. It ships deep learning-powered features in real products [14]. Keras is a high-level neural network library that runs on top of TensorFlow [41]. TensorFlow Decision Forests are a collection of state-of-the-art algorithms for the training, serving and interpretation of Decision Forest models which support classification, regression and ranking [15]. A decision forest is an ensemble of randomly trained decision trees whose prediction is the aggregation of predictions of its decision trees [16], [17]. A decision tree is a hierarchical structure of connected nodes, such that during training, all training data {v} is sent into the tree followed by optimizing parameters of split nodes, so as to optimize a chosen energy function, as shown in Figure 2.

During testing, a split (internal) node applies a test to the input data v and sends it to the appropriate

child iterating the process until a leaf (terminal) node is reached (beige path), as depicted in Figure 3 [16].

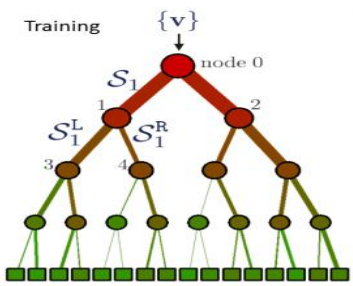


Figure 2. Training of a decision tree [16].

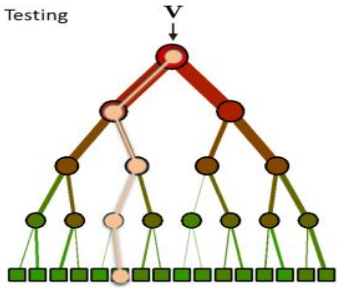


Figure 3. Testing of input data v in a decision tree [16].

Criminisi et al. [16] argue that decision forests compare favourably with respect to other techniques and have led to one of the biggest success stories of computer vision in recent years. They attribute the recent revival of decision forests to the discovery that ensembles of slightly different trees tend to produce much higher accuracy on previously unseen data, a phenomenon known as generalization. According to Rokach [18], decision trees have a high predictive performance for a relatively small computational effort, are able to handle a variety of input data including textual data and scale well to big data as compared to other methods. For experiments in this work, the Gradient-boosted trees and Random Forest algorithms were used.

Boosting is a method for converting a weak learning algorithm into one that achieves arbitrarily high accuracy by sequentially applying weak learners to repeatedly re-weighted versions of the training data [19]. In gradient boosting, the learning procedure consecutively fits new models to provide a more accurate estimate of the response variable with an objective of constructing new base-learners that maximally correlate with the negative gradient of the loss function associated with the whole ensemble [33]. Random forests are a combination of tree predictors, such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [20]. Krauss et al. [19] explain that boosting works by sequentially applying weak learners to repeatedly reweighted versions of the training data whereby after each boosting iteration, misclassified examples have their weights increased and correctly classified examples have their weights decreased. They conclude that each successive classifier focuses on examples that have been hard to classify in the previous steps [19]. Mathematically, an ensemble of trees can be written as:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in F \quad (1)$$

where K is the number of trees, f_k is a function in the functional space F and F is the set of all possible classification and regression trees (CARTs) [21]. The objective function to be optimized is given by:

$$obj(\theta) = \sum_{i}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \omega(f_k) \quad (2)$$

where $\omega(f_k)$ is the complexity of the tree f_k [21].

GloVe embedding –an unsupervised learning algorithm for obtaining vector representations for words– has been extensively used in hate speech detection (e.g. by Gomez et al. [6], Dorris et al. [9], Ong [11] and Badjatiya et al. [8]). Pre-trained embeddings are useful when available training data is limited for data-hungry deep-learning methods [25]. The Universal Sentence Encoder (USE) embedding [26] was used to encode text into high-dimensional vectors for text classification. USE is a model for producing sentence embeddings that demonstrate superior transfer to a number of other NLP tasks as compared to pre-trained word embeddings, such as those produced by Word2vec or GloVe [25]. Word embeddings give a way to use an efficient, dense representation of vectors in which similar words have similar encodings [26]. The USE is trained and optimized for greater-than-word length text, such as sentences, phrases or short paragraphs the input of which is a variable-length English text and the output is a 512 dimensional vector [26]. This encoder differs from word-level embedding models, since it is trained on a number of natural-language prediction tasks that require modeling the meaning of word sequences rather than just individual words [26]. The USE was partially trained with custom text-classification tasks in mind, as shown by Figure 4.

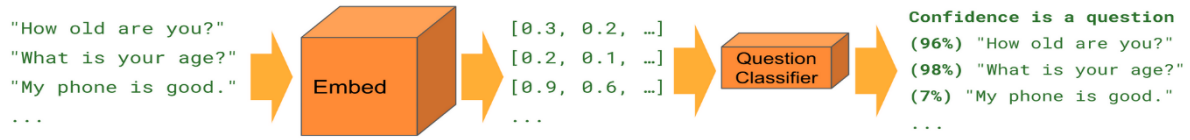


Figure 4. Classification task of the Universal Sentence Encoder (USE) [26].

4. EXPERIMENTAL METHOD

Data whose scope was of political conversations in Kenya was collected from Twitter using Twitter API V2 [27]. For training the models, data was collected from 1st January 2017 to 31st May 2022 whereas for testing on real data, data was collected from 1st August 2022 to 10th August 2022. This period was significant, since it corresponded to the week of the 2022 general elections in Kenya which were held on 9th August 2022. It was believed that during this period, political hate speech could hit a peak. To ensure that relevant data was collected from Twitter, a query was developed whose search criteria included words that had been identified by the National Cohesion and Integration Commission (NCIC) as words that had been used to propagate political hate against specific groups of people in Kenya [28]. This ensured that tweets that were mined were potentially of political hate speech. While tweet data with many attributes including text, images and videos were downloaded, focus was mainly on the text attribute since most hate speech messages on social media are constructed through texts [7] and that text models tend to outperform multi-modal models in detecting hate speech [6]. A total of 46957 records were used for training the model, while 17873 records were used to test it.

Although the context of tweets that were mined focused on hate speech words as defined by the NCIC, it is good to note that not all tweets with these words are necessarily of hate nature. To annotate hate tweets from non-hate tweets, sentiment analysis was performed on the text for each tweet to determine its speech score using the NLTK's Valence Aware Dictionary and Sentiment Reasoner (VADER) [29]. The current version of VADER has the capability of properly handling sentences with punctuation, word-shape, sentiment-laden slang words as modifiers, sentiment-laden emoticons, sentiment-laden initialism and acronyms and utf-8 encoded emoji [30]. Therefore, pre-processing the text data through tokenization or lemmatization was not done before performing sentiment analysis on it with VADER. For each statement, VADER gives probability scores for positive, neutral and negative sentiments all of which must sum to 1. It also gives a compound (average) score for all the scores. A compound score was used to determine sentiments for each tweet text. VADER recommends positive sentiments for compound scores greater than 0.05, neutral for compound scores greater than -0.05, but less than 0.05 and negative sentiments for compound scores of less than -0.05. For the case of this study, all neutral and positive sentiments were considered as not constituting hate speech, while all negative sentiments were considered as potentially constituting hate speech. Thus all tweets with a sentiment compound score of greater than -0.05 were labelled as 0; i.e., "Not_Hate", while those less than or equal to -0.05 were labelled as 1; i.e., "Hate". Figure 5 shows a schema of the method applied in this study.

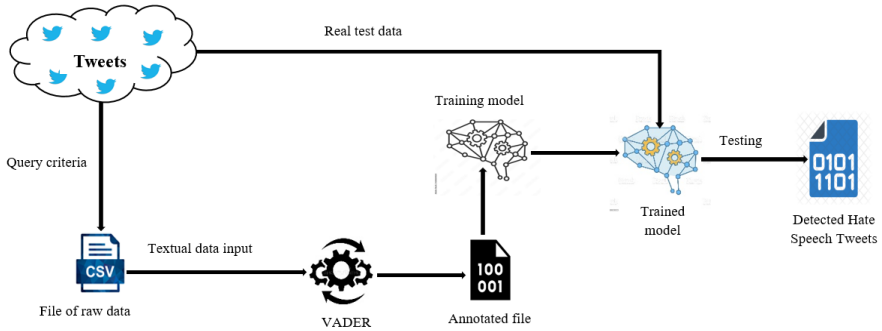


Figure 5. Schema for the method applied in this study.

USE embeddings [26] and Keras TensorFlow Decision Forests (TF-DF) [15] were used to encode text into high-dimensional vectors and train models for text classification, respectively. Default Keras hyper-parameters were applied. Three neural-network models were developed using Keras. In the first model, raw text was first encoded via pre-trained embeddings and then passed to a gradient boosted tree model for classification. In the second and third models, raw text was directly passed to the gradient boosted

trees model and RandomForestModel, respectively. The models were compiled with Google Colab [31] TPU by passing Accuracy, Recall, Precision and AUC metrics. TF-DF automatically detects the best loss for the task; i.e., either classification or regression.

5. RESULTS

Table 1 shows a summary of the results from the experiments.

Table 1. Performance of the three models measured against Accuracy, Recall, Precision and AUC.

Model	Algorithm	Accuracy	Recall	Precision	AUC
Model_1	GradientBoostedTrees with USE embeddings	0.9886	0.9587	0.9831	0.9984
Model_2	GradientBoostedTrees	0.9272	0.6302	1.0000	0.9240
Model_3	RandomForestModel	0.9272	0.6302	1.0000	0.8085

The following graphs show accuracy and loss against number of trees for the models.

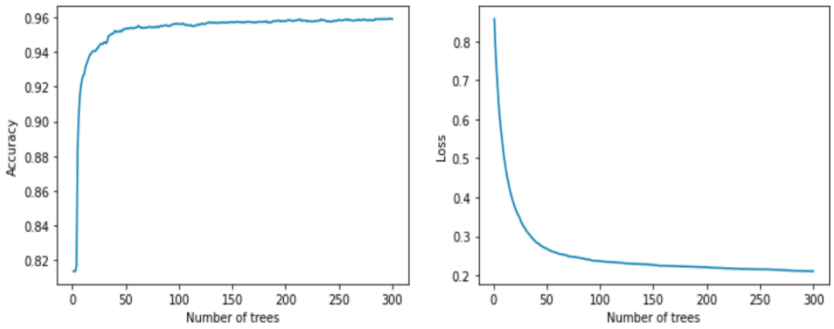


Figure 6. Graph showing accuracy (left) against number of trees and loss against number for model 1.

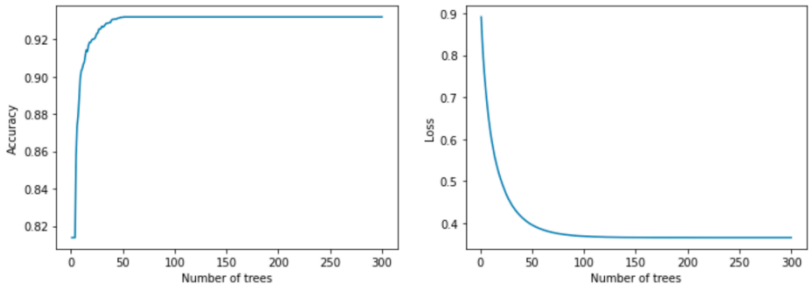


Figure 7. Graph showing accuracy against number of trees (left) and loss against number for model 2.

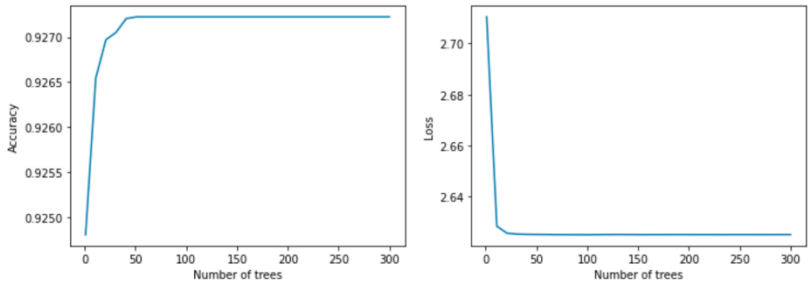


Figure 8. Graph showing accuracy against number of trees (left) and loss against number for model 3.

6. DISCUSSION

From Table 1, it can be observed that model_1; i.e., the GradientBoostedTrees with USE embeddings model has generally performed well according to most metrics. The training logs of accuracy against number of trees and the loss against number of trees also indicate that this model has a superior performance. Training logs show the quality of the model according to the number of trees in the model [32]. When compared with previous studies on detection of hate speech on social-media platforms, the approach of GradientBoostedTrees with USE embeddings still exhibits a superior performance as

demonstrated by Table 2. While experiments for these studies were done in different contexts with different datasets, comparing them may give us an idea on the performance of individual models. An accurate comparison would require replication of the experiments while keeping the dataset constant, but varying the modelling algorithms.

Table 2. Performance comparison of hate speech detection models.

Author	**Best performing technique**	**Metrics for measuring model performance**
Gomez et al. [6]	LSTM + Glove embeddings	F-Score = 0.703, AUC = 0.732 Accuracy = 68.4%
Ong (2019) [11]	BiLSTM-CNN	F1-Score = 0.75
Badjatiya et al. [8]	LSTM, LSTM + Random Embedding + Gradient, Boosted Decision Trees	Precision = 0.930 Recall = 0.930 F1 = 0.930
Dorris et al. [9]	LSTM	Accuracy 90.82%
This research	GradientBoostedTrees with USE Embeddings	Accuracy = 98.86% Recall = 0.9587 Precision = 0.9831 AUC = 0.9984

The graphs in Figures 6–8 also show that the GradientBoostedTrees with USE embeddings model suffers less loss in comparison to the GradientBoostedTrees model and the RandomForestModel. The purpose of this research was to propose a model for detecting political hate speech propagated through social-media platforms. Considering the performance of the GradientBoostedTrees with USE embeddings model, this model is proposed for detecting hate speech on social-media platforms.

However, we should remember that this model works only on textual data from social-media discourses. We are aware of other methods through which hate speech is transmitted over social-media platforms. As discussed earlier, these methods include obfuscation of hate messages by distorting spelling of words, hiding hate messages in images and using sarcasm through text or audio-visual methods. There is a need to continue searching for models that can help the society detect hate speech propagated through these methods. Although Gomez et al. [6] found out that text models outperform multi-modal models, we should not stop research on multi-modal methods. If we do so, multi-modal hate speech will become a key conduit of hate speech propagation on social media. Finally, there is a need for the research community to invest in replication of already proposed models, so as to test their viability and applicability.

6. CONCLUSION

Although propagation of hate-speech over social-media platform is relatively a recent phenomenon, several researchers have proposed various approaches for tackling this problem with varying success. The purpose of this research was to propose a model for detecting political hate speech in Kenya propagated through social-media platforms. Experimenting with data from Twitter, it was found out that the GradientBoostedTrees with USE embeddings model exhibited a superior performance as compared to other models previously proposed in literature. However, actualizing this model at production level may pose some challenge, since it may be difficult to apply it in real-time detection of hate speech, considering that it requires massive computational resources for training, compiling and testing the model. For this research, Colab's TPU was used to train and test the model. Currently, such a resource may not be available around the clock for industrial deployment of such a model. Secondly, the model depends heavily on hate-speech words as defined by the NCIC. These words have their original meanings which are not necessarily of hate-speech nature. Similarly, as language evolves, the meaning of the NCIC's hate speech words is bound to evolve too. Therefore, it is not guaranteed that those words will always imply hate speech. This challenge can easily generate false positive results.

FUTURE WORK

A replication of experiments in this research with an objective of achieving results that can be generalized is necessary.

ACKNOWLEDGEMENTS

The author would like to thank Twitter, Inc. for allowing the study to access data from their databases for research purpose, Google LLC for availing their TPU computing resources for this research and VADER developers for making their tool open source.

REFERENCES

[1] U. Nations, "Understanding Hate Speech," [Online], Available: https://www.un.org/en/hate-speech/understanding-hate-speech/what-is-hate-speech, 20 Oct. 2022.

[2] U. Nations, "Hate Speech," [Online], Available: https://www.un.org/en/hate-speech/impact-and-prevention/why-tackle-hate-speech, 20 Oct. 2022.

[3] E. Ombui, L. Muchemi and P. Wagacha, "Hate Speech Detection in Code-switched Text Messages," Proc. of the 3rd IEEE Int. Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp. 1–6, Ankara, Turkey, 2019.

[4] F. Poletto, V. Basile, M. Sanguinetti, C. Bosco and V. Patti, "Resources and Benchmark Corpora for Hate Speech Detection: A Systematic Review," Language Resources and Evaluation, vol. 55, no. 2, pp. 477–523, 2021.

[5] U. Nations, "Hate Speech," [Online], Available: https://www.un.org/en/hate-speech/impact-and-prevention/challenges-of-tracking-hate, 20 Oct. 2022.

[6] R. Gomez, J. Gibert, L. Gomez and D. Karatzas, "Exploring Hate Speech Detection in Multimodal Publications," Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV), pp. 1459-1467, 2020.

[7] N. S. Mullah and W. M. N. W. Zainon, "Advances in Machine Learning Algorithms for Hate Speech Detection in Social Media: A Review," IEEE Access, vol. 9, DOI: 10.1109/ACCESS.2021.3089515, 2021.

[8] P. Badjatiya, S. Gupta, M. Gupta and V. Varma, "Deep Learning for Hate Speech Detection in Tweets," Proc. of the 26th Int. Conf. on World Wide Web Companion, pp. 759–760, DOI:10.1145/3041021.3054223, 2017.

[9] W. Dorris, R. Hu, N. Vishwamitra, F. Luo and M. Costello, "Towards Automatic Detection and Explanation of Hate Speech and Offensive Language," Proc. of the 6th Int. Workshop on Security and Privacy Analytics, pp. 23–29, DOI: 10.1145/3375708.3380312, 2020.

[10] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] R. Ong, "Offensive Language Analysis Using Deep Learning Architecture," arXiv: 1903.05280, DOI: 10.48550/arXiv.1903.05280, 2019.

[12] Github, "Understanding LSTM Networks," [Online], Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/, 21 Oct. 2022.

[13] F. Chollet, "Keras," [Online], Available: https://keras.io/, 21 Oct. 2022.

[14] F. Chollet, "Introduction to Keras for Engineers," [Online], Available: https://keras.io/gettingstarted/ intro to keras for engineers/, 21 Oct. 2022.

[15] Google, "Tensorflow Decision Forests," [Online], Available: https://www.tensorflow.org/decision forests, 20 Oct. 2022.

[16] A. Criminisi, J. Shotton and E. Konukoglu, "Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-supervised Learning [Internet]," Microsoft Research, MSR-TR-2011-114, 2011.

[17] G. Developers, "Decision Forests," [Online], Available: https://developers.google.com/machine-learning/decision-forests/intro-to-decision-forests-real, 20 Oct. 2022.

[18] L. Rokach, "Decision Forest: Twenty Years of Research," Information Fusion, vol. 27, pp. 111–125, 2016.

[19] C. Krauss, X. A. Do and N. Huck, "Deep Neural Networks, Gradient-boosted Trees, Random Forests: Statistical Arbitrage on the S&P 500," Europ. J. of Operat. Research, vol. 259, no. 2, pp. 689–702, 2017.

[20] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[21] Xgboost Developers, "Introduction to Boosted Trees," [Online], Available: https://xgboost.readthedocs. io/en/stable/tutorials/model.html, 20 Oct. 2022.

[22] Google, "Open Sourcing Bert: State-of-the-art Pre-training for Natural Language Processing," [Online], Available: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html, 20 Oct. 2022.

[23] M. Mozafari, R. Farahbakhsh and N. Crespi, "A BERT-based Transfer Learning Approach for Hate Speech Detection in Online Social Media," Proc. of the Int. Conf. on Complex Networks and Their Applications, Computational Intelligence Book Series, vol. 881, pp. 928–940, Springer, 2019.

[24] A. Velankar, H. Patil, A. Gore, S. Salunke and R. Joshi, "L3Cube-mahahate: A Tweet-based Marathi Hate Speech Detection Dataset and BERT Models," Proc. of the 3rd Workshop on Threat, Aggression and

Cyberbullying (TRAC 2022), pp. 1-9, Gyeongju, Republic of Korea, 2022.
[25] D. Cer, Y. Yang, S.-Y. Kong et al., "Universal Sentence Encoder," arXiv: 1803.11175, 2018.
[26] T. Hub, "Universal-sentence-encoder," [Online], Available: https://tfhub.dev/google/universal-sentence-encoder/4, 20 Oct. 2022.
[27] Twitter, "Tweet Downloader," [Online], Available: https://developer.twitter.com/apitools/downloader, 20 Oct. 2022.
[28] N. Cohesion and I. Commission, "Hatelex: A Lexicon of Hate Speech Terms in Kenya," Nairobi, Tech. Rep., 2022.
[29] C. Hutto and E. Gilbert, "Vader: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," Proc. of the Int. AAAI Conf. on Web and Social Media, vol. 8, no. 1, pp. 216–225, 2014.
[30] GitHub, "Vadersentiment," [Online], Available: https://github.com/cjhutto/vaderSentiment, Oct. 2022.
[31] Google, "Welcome to Colaboratory," [Online], Available: https://colab.research.google.com, Oct. 2022.
[32] Google, "Build, Train and Evaluate Models with Tensorflow Decision Forests," [Online], Available: https://www.tensorflow.org/decision forests/tutorials/beginner colab, 20 Oct. 2022.
[33] A. Natekin and A. Knoll, "Gradient Boosting Machines: A Tutorial," Frontiers in Neurorobotics, vol. 7, p. 21, 2013.
[34] S. Khan, A. Kamal, M. Fazil et al., "HCovBi-Caps: Hate Speech Detection Using Convolutional and Bi-directional Gated Recurrent Unit with Capsule Network," IEEE Access, vol. 10, pp. 7881–7894, 2022.
[35] B. Vidgen, T. Thrush, Z. Waseem and D. Kiela, "Learning from the Worst: Dynamically Generated Datasets to Improve Online Hate Detection," Proc. of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th Int. Joint Conf. on Natural Language Processing, vol. 1: Long Papers, pp. 1667–1682, DOI: 10.18653/v1/2021.acl-long.132, 2021.
[36] S. Khan, M. Fazil, V. K. Sejwal, M. A. Alshara, R. M. Alotaibi and Kamal, "BICHAT: BiLSTM with Deep CNN and Hierarchical Attention for Hate Speech Detection," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 7, pp. 4335 – 4344, 2022.
[37] C. Koutlis, M. Schinas and S. Papadopoulos, "MemeTector: Enforcing Deep Focus for Meme Detection," arXiv: 2205.13268, DOI: 10.48550/arXiv.2205.13268, 2022.
[38] A. Aggarwal, V. Sharma, A. Trivedi et al., "Two-way Feature Extraction Using Sequential and Multimodal Approach for Hateful Meme Classification," Complexity, vol. 2021, pp. 1–7, 2021.
[39] IBM, "AI *vs.* Machine Learning *vs.* Deep Learning *vs.* Neural Networks: What's the Difference?" [Online], Available: https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks, 30 Jan. 2023.
[40] O. Sagi and L. Rokach, "Ensemble Learning: A Survey," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 8, no. 4, p. e1249, 2018.
[41] V. Zocca, G. Spacagna, D. Slater and P. Roelants, Python Deep Learning, 2nd Edition, ISBN: 978-1789348460, Packt Publishing Ltd, 2017.

ملخص البحث:

يشكّل الكشْف عن خِطاب الكراهية والسّيطرة على انتشاره عبر منّصات التّواصل الاجتماعي تحدّياً كبيراً. وتتفاقم هذه المشكلة بفعل التّدفّق فائق السرعة، وتوافر الجمهور، وديمومة المعلومات على منصّات التّواصل الاجتماعي. ويهدف هذا البحث الى اقتراح نموذج يمكن استخدامه للكشْف عن خِطاب الكراهية السّياسي الذي ينتشر على منصّات التّواصل الاجتماعي في كينيا.

وقد تمّ تطوير ثلاثة نماذج لهذا الغرض باستخدام نُصوصٍ مأخوذة من تويتر. وبمقارنة نتائج النّماذج الثلاثة المطوّرة، اتّضح أنّ نموذج (GBT) مع التّضمين العامّ للجُمل (USE) قد حقّق أفضل النّتائج بدقّةٍ بلغت (99.86%)، واستعادةٍ بلغت (0.9587)، وضبطٍ مقداره (0.9831)، بينما كانت المساحة المحصورة تحت المنحنى لهذا النّموذج (0.9984). وعليه، فإنّ هذا النّموذج يمكن استخدامه للكشْف عن خِطاب الكراهية الذي ينتشر على منصّات التّواصل الاجتماعي.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 01, March 2023.

ORTHOGONAL REGRESSED STEEPEST DESCENT DEEP PERCEPTIVE NEURAL LEARNING FOR IOT-AWARE SECURED BIG DATA COMMUNICATION

S. L. Swapna[1*] and V. Saravanan[2]

(Received: 30-Nov.-2022, Revised: 21-Jan.-2023 and 15-Feb.-2023, Accepted: 19-Feb.-2023)

ABSTRACT

The Internet of Things (IoT) is a collection of interconnected intelligent devices that exist within the larger network known as the Internet. With the increasing popularity of IoT devices, massive data is generated day by day. The collected data needs to be continuously uploaded to the cloud server. Besides, the transmission of data in the cloud environment is performed via the Internet, which faces numerous threats. However, the security issue always lacks effective big data communication. Therefore, a novel technique called Orthogonal Regressed Steepest Descent Deep Structured Perceptive Neural Learning based Secured Data Communication (ORSDDSPNL-SDC) is introduced with higher accuracy and lesser time consumption. The ORSDDSPNL-SDC technique comprises three phases; namely, registration, user authentication, and secure data communication. In the ORSDDSPNL-SDC technique, the registration phase is carried out for creating a new ID and a password for each user in the cloud. The IoT device's data is then sent to a cloud server by the cloud user for storage. After that, the orthogonal regressed steepest descent multilayer deep perceptive neural learning is applied to examine the user_ ID with already registered ID based on Szymkiewicz–Simpson coefficient. Then, the Maxout activation function is to classify the user as authorized or unauthorized. Finally, the steepest descent function is applied for minimizing the classification error and increasing the classification accuracy. In this way, the authorized or unauthorized user is identified. Then, the secured communication is performed with the authorized cloud users. Experimental evaluation is carried out on the factors, such as classification accuracy, classification time and error rate, and space complexity for several users. The qualitative results and discussion indicate that the proposed ORSDDSPNL-SDC offers an elevated performance by achieving a higher classification accuracy and a minimum error as well as computation time when compared to the existing methods.

KEYWORDS

Big data, IoT, Multilayer deep perceptive neural learning, Orthogonal regression analysis, Szymkiewicz–Simpson coefficient, Maxout activation function, Steepest descent function.

1. INTRODUCTION

The Internet of Things (IoT) has gained broad acceptance and increased in various aspects of human life. Heterogeneous big data security is a great challenge in IoT due to the rapid growth of data. Therefore, novel security approaches are needed for data communication and outsourcing to storage systems through the authorized entity. Hence, an IoT and deep learning-based secure data analytics system is proposed introduced to address this security issue.

A Homomorphic Block-Ring Security System (HBRSS) was developed in [1] for enhancing the security of data communication. HBRSS brings higher security and more comprehensive performance compared with some mainstream security systems. HBRSS is proposed for high security data transmission and data processing in public networks and mistrusted cloud environments. However, a higher security level was unable to achieve. A trusted and Collaborative Framework for Deep Learning enabled IoT was designed in [2] to enhance data transmission and computation security. Though the framework minimizes the overhead, time complexity was not reduced.

Deep learning and IoT-based data processing frameworks were introduced in [3] in various security concerns to include malicious code tracking, intrusion, privacy issues, vulnerability detection, and fault diagnosis.

In [4] a reliable lightweight authentication scheme for IoT-based secure data sharing was developed. But, the machine learning technique was not applied for improving the authentication accuracy. In [5],

1. S. L. Swapna (Corresponding Author) is with Department of Computer Science, Hindusthan College of Arts and Science (Autonomous), Coimbatore, India. Email: swapnamartin2003@gmail.com
2. V. Saravanan is with Department of Information Technology, Hindusthan College of Arts and Science (Autonomous), Coimbatore, India. Email: vsreesaran@gmail.com

a secure authentication protocol for cloud big data was introduced to reduce computing power. The storage costs were not lowered. A lightweight authentication technique was developed in [6] for ensuring device and cloud server security. But, accurate authentication was not performed with minimum time.

A lightweight identity-based authenticated data-sharing protocol was designed in [7] for enhancing the security of data distribution between physical devices and clients. But, the designed protocol failed to evaluate in a real-world setting. AI-enabled lightweight, a secure communication method for an IoMT called ASCP-IoMT was developed in [8] to improve accuracy. But, it failed to consider the functionality of features in the presented scheme.

In [9], a lightweight and secure communication technique was developed for safe data transmission among healthcare infrastructure devices. However, the registration process was completed offline. An ultra-lightweight device-to-device secure transmission protocol was introduced in [10] to protect data transmission. However, when dealing with a large number of devices, it was unable to guarantee the complete security of the IoT environment. In [11], two optimization schemes were designed to prevent user access from leaking data while also protecting users' privacy. The designed schemes have higher computation costs.

The following are the research contributions of the ORSDDSPNL-SDC technique.

- A novel privacy-preserving technique called ORSDDSPNL-SDC is introduced for secure big data transmission by identifying the authorized user with a better accuracy and in less time.
- First, the registered cloud users put their information with the cloud service and receive a user _ID and a password each for further processing. This ID is used for identifying the authorized user with minimum time as well as storage overhead.
- To improve secure big data transmission between the server and the users, orthogonal regressed steepest descent multilayer deep perceptive neural learning is applied in ORSDDSPNL-SDC to analyze the received user ID with registered user ID through the Szymkiewicz–Simpson coefficient. The Szymkiewicz–Simpson coefficient is a mathematical function used to match the ID of the user for identifying authorized users. The Maxout activation function identifies the user as authorized or unauthorized based on classification results. This helps improve classification accuracy.
- Finally, the steepest descent function is used to reduce the error rate with aid of the weight updating rule.
- A series of experimental assessments is carried out to quantify the proposed ORSDDSPNL-SDC method in comparison to existing algorithms and performance metrics.

The rest of the paper is organized as follows. Section two presents a literature review on the IoT security of application domains. The third section describes the proposed ORSDDSPNL-SDC for IoT security. Section four provides the experimental settings and dataset description. Section five presents the outcomes and discussion generated by various performance indicators and the paper is concluded in Section six.

2. RELATED WORKS

Big data security is a major concern due to its critical role in the widespread adoption of cloud architectures. In [12], a case-based security-by-design framework for big data deployment over cloud computing was developed. The study's findings show the effectiveness of raising security awareness in cloud-based data environments.

Multifactor authentication and lightweight cryptography encryption methods were developed in [13] to protect big data systems by using a cloud-enabled IoT environment. But, those methods failed to perform mutual authentication between gateway devices and IoT devices. To identify the genuine or fraudulent user, a deep learning-based IoT data analytics method was devised [14]. But, despite employing larger datasets, it was unable to reduce the time and cost limits.

For safe big-data transmission, the Data-centric Authentication technique was introduced in [15]. However, neither the time nor the complexity of the storage was reduced. Secure Authentication and Data Sharing in Cloud (SADS-Cloud) were introduced in [16] for data security with big data. However,

they failed to speed up the operations of secure authentication and data sharing in the cloud.

According to [17], MSCryptoNet is a model based on multi-scheme fully homomorphic encryption in the privacy-preserving scenario that allows for scalable execution and conversion of the trained neural network. MSCryptoNet was claimed to be secure as a deep learning-based privacy-preserving scheme over aggregated encrypted data. Furthermore, it was claimed that the MSCryptoNet does not require any correspondence between data providers and the cloud server to provide privacy-preserving predictions. However, there is no relevant evidence in the literature to support the second claim.

An enhanced Diffie Hellman when combined with Elliptic Curve (E-ECDH) was developed in [18] for secure and lightweight communications of connected devices in IoT. However, it failed to improve the security of IoT by considering applications in different domains. A Chebyshev Chaotic Map-based lightweight multi-server authentication method was developed in [19] for secure data transmission with minimum computation cost. However, the storage overhead was not reduced.

In [20], a lightweight and privacy-aware fine-grained access control method for secure IoT-oriented health data transmission was developed. However, authentication-based access control was not implemented. Privacy-preserving hierarchical fuzzy neural network (PPHFNN) was developed in [21] for improving the security of heterogeneous big data. However, the performance of the accuracy level was not improved.

A three-factor user access control mechanism for data usage in the IoT environment is presented in [22]. Despite claims that the suggested approach is secure, it was unclear what kinds of threats the method handled. By combining attribute-based encryption with edge computing, [23] implemented a mechanism for a healthcare IoT system. It was asserted that the suggested mechanism offers a reliable, adaptable, secured access control with data verification and permits data consumers to take use of the lightweight decryption. In [24], a crypto-deep neural network cloud security (CDNNCS) method for increasing cloud user trust was developed. However, the other deep learning architectures were not implemented for the dynamic behavior of communication in the cloud environment. Multifactor authentication and lightweight cryptography encryption methods were developed in [25] based on cloud-enabled IoT environments to protect big data systems. However, mutual authentication between devices and IoT devices was not performed.

3. METHODOLOGY

The Internet of Things (IoT) system is composed of several internetworked smart equipment that shares information *via* the world wide web. Such self-directed devices are dispersed in specific fields to sense and collect data. The data collected from IoT sensors is transferred to cloud servers for storage and further processing in IoT cloud-based systems. The data is then accessed by authorized users from the cloud server. However, security concerns are growing in conjunction with the widespread adoption of IoT-based infrastructure. As a result, there is an urgent need to propose novel security mechanisms to protect IoT systems from potential threats. Based on this motivation, a novel technique called ORSDDSPNL-SDC is introduced. The ORSDDSPNL-SDC technique accurately identifies the authorized or unauthorized user to get the data from the cloud service through orthogonal regressed steepest descent multilayer deep perceptive neural learning. The advantage of multilayer deep perceptive neural learning is to provide better results with a large volume of big data.

The architecture diagram of the proposed ORSDDSPNL-SDC technique for secure big data communication is shown in Figure 1. The proposed technique comprises two types of entities; namely, cloud users '$CU = CU_1, CU_2, \ldots, CU_n$ who want to store their big data '$D = D_1, D_2, \ldots., D_n$' collected from various IoT devices. The proposed ORSDDSPNL-SDC technique includes three major processes; namely, registration, user authentication and secure data communication.

The IoT device allows the user to collect the data simply using the different sensors installed at various locations. When the users want to store their collected data from IoT devices on the cloud server, they first perform the registration process. The user has to fill up the registration form which is provided by the cloud server. It contains information about the user, like name, date of birth, age, gender, mobile number and so on. After filling out the registration form, the user presses the submit button. The cloud server stores the user details in its database and creates the new user_ID.

After successful registration, the user stores the data in the cloud server database for further processing. Whenever the users want to access their desired data from the cloud-server database, they first verify their authenticity. The proposed ORSDDSPNL-SDC technique uses the orthogonal regressed steepest descent multilayer deep perceptive neural learning for identifying the authorized or unauthorized user through classification.

An orthogonal regressed steepest descent multilayer deep perceptive neural learning is a family of machine learning methods that work based on artificial neural networks. The word deep learning refers to the use of numerous layers utilized in the network for learning the given input. The orthogonal regression is applied to deep learning for verifying the authenticity of the user by analyzing the current ID and the already stored ID in the server database based on the Szymkiewicz–Simpson coefficient. The Szymkiewicz–Simpson coefficient is a similarity function that is used to match the IDs of the cloud users. If the two IDs get matched, the max-out activation function returns the output as an authorized user or the unauthorized user. The cloud server offers the user the requested data for enhancing secure communication after identifying him/her as an authorized user. Otherwise, the cloud server denies access to unauthorized users. Secure communication between cloud users and servers is thus accomplished. A brief description of the proposed ORSDDSPNL-SDC technique with different processes is given below.

3.1 New User Registration

The registration phase is the initial step in the proposed ORSDDSPNL-SDC technology. The users must provide the necessary identification during the registration step to register their information in the server database. The user has to fill out a registration form which contains information about username, date of birth, age, gender, mobile number, mail_ID and so on.

Figure 2 exhibits the flow process of new user registration. First, the users enter their personal information in the registration form. The user has to give a valid username at the time of registration. The server checks the created username against the availability of that username in its database. If the username is not matched with the existing username in the server database, then the server sends an error and try again message. Otherwise, a new user ID is generated and stored in the server database for later use. Following registration, the users uploads their data '$D = D_1, D_2, \ldots, D_n$' collected from various IoT devices into the server database.

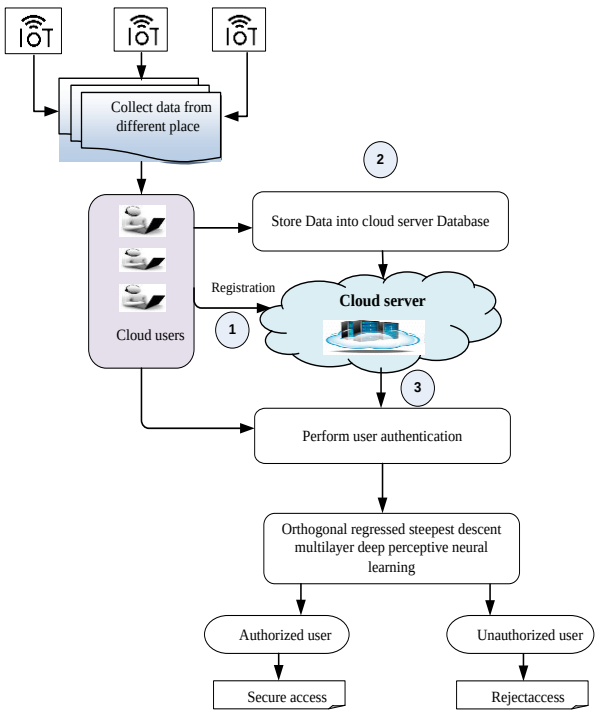


Figure 1. Architecture diagram of the proposed ORSDDSPNL-SDC technique.

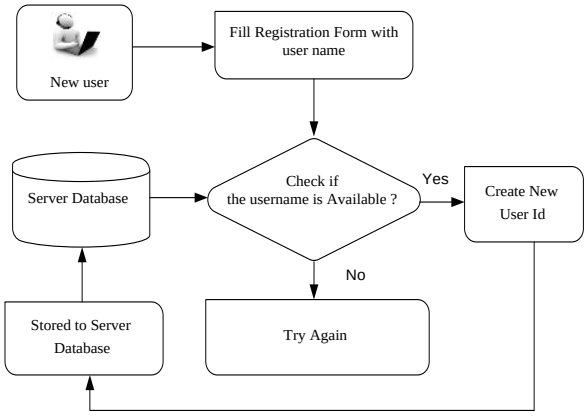


Figure 2. Flow process of new user registration.

The input dataset was separated into two parts; namely, the training and testing. Most cloud users (70%) were used for training and the remaining cloud users (30%) were taken for testing. Here, MHEALTH (Mobile HEALTH) dataset is considered for collecting the data. The different sensors (i.e., IoT devices) are located on the candidate's chest, right wrist and left ankle for recordings of body motion and vital indications while performing physical activities. The activities are standing still, sitting and relaxing, lying down, walking, climbing stairs, waist bends forward, frontal elevation of arms, knees bending, cycling, jogging, running, jumping front and back, respectively. The big data collected from IoT devices is sent to a cloud data center for further processing.

3.2 Orthogonal Regressed Steepest Descent Multilayer Deep Perceptive Neural Learning-based User Authentication

Whenever the users access their data from the cloud server database, they first verify authenticity for guaranteeing secure communication. User authentication is the process of verifying the identity of the user. The proposed ORSDDSPNL-SDC technique uses the deep learning technique called orthogonal regressed steepest descent multilayer deep perceptive neural learning for verifying the authenticity of the user by classifying the user as an authorized user or as an unauthorized user. Orthogonal regressed steepest descent multilayer deep perceptive neural learning is a fully connected feed-forward artificial neural network. A multilayer perceptron includes at least three layers of nodes such as an input layer, a hidden layer and an output layer.

Figure 3 reveals the schematic structure of the deep multilayer perceptive learning technique for user authentication. The input layer is initially given the number of cloud users. The network has many layers, including an input layer, hidden layers and an output layer. Each layer in deep learning learns the given input and transforms it into the next layer.

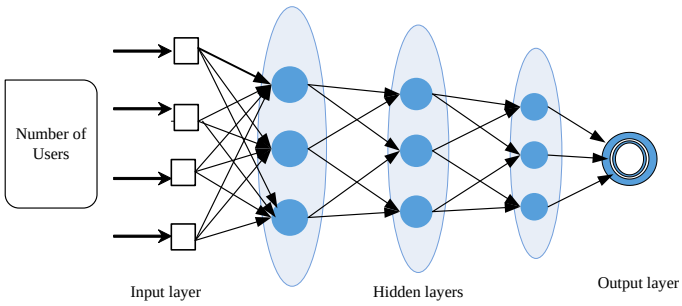


Figure 3. The orthogonal regressed steepest descent multilayer deep perceptive neural learning architecture.

To form the neural network architecture, the nodes between one layer and another are linked in a feed-forward way by regulating weights. The input layer accepts the cloud users $CU = CU_1, CU_2, \dots, CU_n$.

$$X_{input}(t) = B + \left[\sum_{i=1}^{n} CU_i * \varphi_{i\_h}\right] \quad (1)$$

where, $X_{input}(t)$ indicates the input unit, φ_{ih}denotes a controlling weight between the input layer and the hidden layer, CU_i indicates a cloud user and B denotes a bias in which the integer value stored is +1. The input is sent to the first hidden layer.

When a registered cloud user desires to retrieve big data from the cloud service, he/she must first log in to the cloud server and enter a valid ID on his/her log-in page. Then, the server identifies the authorized user or unauthorized user with the help of orthogonal regression.

Orthogonal regression is a statistical technique for estimating relationships and determining the best fit between two variables, such as the currently received user ID and the already stored ID. The Szymkiewicz–Simpson coefficient is a similarity function that is used to calculate the degree of overlap between two finite sets; namely, the currently entered user ID and the already stored ID. The Szymkiewicz–Simpson coefficient is used to identify the

$$Cff_{ss} = \frac{|RID \cap RegID|}{|S_{RID}||RegID|} \quad (2)$$

where, Cff_{ss} represents the Szymkiewicz–Simpson coefficient, RID indicates a received ID and $RegID$

indicates a registered ID, $|S_{RID}|$ denotes a set containing a string of received IDs, $|RegID|$ denotes a set containing a string of registered IDs, $RID \cap RegID$ designates mutual dependence between the two IDs.

In the second hidden layer, Maxout activation is applied for verifying the similarity coefficient value. The activation function of a node in the layer defines the output of that node based on a given set of inputs. Maxout activation finds the maximum value of the similarity coefficient.

$$\beta = \begin{cases} 1; & argmax\ Cff_{ss} \\ 0; & otherwise \end{cases} \tag{3}$$

where, β denotes the Maxout activation function, the activation function (β) returns '1' when the two IDs get matched and the user is classified as an authorized user. The activation function (β) returns '0', which means that the two IDs are not matched and the user is classified as an unauthorized user. The hidden layer output is formulated as given below,

$$Z = \left(\sum_{i=1}^{n}\sum_{i=1}^{n} CU_i * \varphi_{i\_h}\right) + [Z' * \varphi_h] \tag{4}$$

where, 'Z' indicates the output of the hidden layer, 'φ_{i_h}' denotes the regulating weight between the input and the hidden unit, φ_h indicates the regulating weight of the hidden unit and Z' denotes an output of the previously hidden layer.

$$W = \varphi_{h\_o}\, Z \tag{5}$$

where, 'W symbolizes the final results of the output layer, φ_{h_o} represents the weight between the hidden layer and the output layer and Z denotes the output of the hidden layer. After classification, the degree of training error in an output node is calculated as follows:

$$T\varepsilon = \frac{1}{2}(T_o - W)^2 \tag{6}$$

where, $T\varepsilon$ denotes a training error, T_o indicates a target output value and W denotes an output produced by the multilayer perceptron. Applying the steepest descent function leads to minimize the error in the entire output by changing the weight between the layers. There is a mathematical steepest descent function that is used to minimize the error by using the update rule.

$$\Delta\varphi = -\gamma\ \frac{\partial T\varepsilon}{\partial \varphi} W \tag{7}$$

where, $\Delta\varphi$ updating the weight, γ indicates a learning rate, $T\varepsilon$ denotes an error and W indicates an output of classification result. In this way, accurate classification of the authorized and unauthorized users is determined with minimum error. Algorithm 1 describes the step-by-step process of secure data communication using Orthogonal Regressed Steepest Descent Deep Structured Perceptive Neural Learning.

The users first register their information with the cloud service. The cloud server then generates a new user ID for accessing the service. After receiving the new ID, the user stores the data collected from the IoT device in a cloud server. Every time a user accesses data from the cloud server, it first confirms the user's ID. The Szymkiewicz–Simpson coefficient is applied for verifying the user ID. If the two IDs get matched, the max-out activation function returns '1' and the user is classified as authorized. Otherwise, the Maxout activation function returns '0' and the user is classified as unauthorized. Finally, the training error of classification is minimized by updating the weight between the layers using the steepest descent function. Then, the cloud server allows for accessing the data to the authorized user and denied access to the unauthorized user. In this way, secure big data communication is performed.

4. EXPERIMENTAL SETTING

In this section, the CloudSim simulator is used to perform an empirical analysis of the proposed ORSDDSPNL-SDC technique as well as the existing HBRSS [1], TCFDL [2] and deep learning and IoT-based data processing frameworks [3]. Secure big data communication is performed using the MHealth dataset taken from https://www.kaggle.com/datasets/gaurav2022/mobile-health. The MHEALTH (Mobile HEALTH) dataset comprises recordings of 10 individuals' body motions and vital signs as they engaged in twelve different physical activities. Sensors placed on the subject's chest, right

Algorithm 1: Orthogonal Regressed Steepest Descent Deep Structured Perceptive Neural Learning based Secured Data Communication

Input: Dataset 'DS', IoT device '$S = S_1, S_2, \dots, S_n$', cloud user's '$CU = CU_1, CU_2, \dots, CU_n$', big data '$D = D_1, D_2, \dots., D_n$', cloud server '$CS$'
Output: Improve the secure communication

Begin
Step 1: For each user
Step 2: Fill in the details in the registration form
Step 3: CS creates a new user_ID
Step 4: CU stores the data on a cloud server
Step 5: End for
Step 6: If CU accesses data from the cloud then
Step 7: CS verifies the authenticity of the user
Step 8: End if
Step 9: User login to the system with User_ID
Step 10: server uses the Szymkiewicz–Simpson coefficient 'Cff_{ss}'
Step 11: Apply max out activation function
Step 12: $\boldsymbol{if}$ $(argmax\ Cff_{ss})$ then
Step 13: β returns '1'
Step 14: the user is classified as authorized
Step 15: Secure data access
Step 16: else
Step 17: β returns '0'
Step 18: The user is classified as unauthorized
Step 19: Denied access
Step 20: Compute error rate '$T\varepsilon$'
Step 21: Apply the steepest descent to minimize the error
Step 22: update the weight '$\Delta\varphi$'
End

wrist and left ankle are utilized to track the acceleration, rate of turn and magnetic field orientation of various body parts. The sensor, which is placed on the chest, may also record a 2-lead ECG to check for various arrhythmias and record the heartbeat. Each sensing modality is recorded on a sampling rate of 50 Hz that is focused enough to capture human activity. All sessions were recorded with a video camera. The dataset is found to simplify general behaviors of the day-by-day living, given the variety of body parts involved in each one (e.g. the frontal elevation of arms *vs.* knees bending), the intensity of the actions (e.g. cycling *vs.* sitting and relaxing) as well as the execution speed or dynamicity (e.g. running *vs.* standing still). The dataset summary, such as 12 activities, 3 sensor devices and 10 subjects, is considered. The dataset consists of 14 attributes (i.e., features) and 12, 15,745 instances (i.e., samples).

Table 1. Attributes' descriptions.

S. No	Attributes	Description
1	alx	Acceleration from the left-ankle sensor (X-axis)
2	aly	Acceleration from the left-ankle sensor (Y-axis)
3	alz	Acceleration from the left-ankle sensor (Z-axis)
4	glx	Gyro from the left-ankle sensor (X-axis)
5	gly	Gyro from the left-ankle sensor (Y-axis)
6	glz	Gyro from the left-ankle sensor (Z-axis)
7	arx	Acceleration from the right-ankle sensor (X-axis)
8	ary	Acceleration from the right-ankle sensor (Y-axis)
9	arz	Acceleration from the right-ankle sensor (Z-axis)
10	grx	Gyro from the right-ankle sensor (X-axis)
11	gry	Gyro from the right-ankle sensor (Y-axis)
12	grz	Gyro from the right-ankle sensor (Z-axis)
13	Activity	Corresponding activity
14	Subject	Volunteer subjects (1 – 9)

The ten volunteers generate a lot of data (i.e., instances) and the data is communicated to the authorized entity. The hyperparameter tuning of the ORSDDSPNL-SDC technique is considered such that the value

of learning rate used is 0.1, the number of hidden layers is 3, the number epochs is 10, the weight is 1.0 and the bias is 0.1 in the ranges of a network trained, 10 cross-validations are used to solve the overfitting/underfitting, imbalance problems. The steepest descent is a first-order iterative optimization algorithm to minimize the loss as quickly as possible.

5. RESULTS AND DISCUSSION

In this section, the performances of the proposed ORSDDSPNL-SDC technique and existing HBRSS [1], TCFDL [2] and deep learning and IoT-based data processing frameworks [3] are discussed with different metrics, such as classification accuracy, error rate and classification time and space complexity. The performance in terms of these parameters is analyzed with the help of a table and a graphical representation.

Classification accuracy: It is calculated as the proportion of cloud users who are correctly classified as authorized or unauthorized users to the total cloud users. The measure of accuracy is mathematically expressed as follows:

$$CA = \left(\sum_{i=1}^{n} \frac{CU_{AC}}{CU_i}\right) * 100 \quad (8)$$

where, CA denotes the classification accuracy and is estimated based on the number of cloud users 'CU_i' and the number of cloud users accurately classified 'CU_{AC}' in the simulation. It is expressed as a percentage (%).

Error rate: It is measured as the ratio of cloud users that are incorrectly classified to the total number of cloud users taken as input. This error rate during the classification is expressed as follows:

$$ER = \left[\sum_{i=1}^{n} \frac{CU_{wC}}{CU_i}\right] * 100 \quad (9)$$

where, 'ER' denotes the error rate, CU_{wC} denotes the cloud users wrongly classified and CU_i represents the number of cloud users. It is measured in terms of a percentage (%).

Classification time: It is defined as the amount of time consumed by the algorithm for classifying authorized and unauthorized users to perform secure communication. The following is a mathematical definition of the classification time:

$$CT = \sum_{i=1}^{n} CU_i * Time\ [classification] \quad (10)$$

where, CT indicates the classification time, CU_i represents the number of cloud users and *time* $[classification]$ denotes the time consumed in classification. It is quantified in milliseconds (ms).

Storage overhead: It is defined as how much memory the algorithm uses to classify authorized and unauthorized users when performing secure communication. The total amount of space consumed is calculated as follows:

$$SO = \sum_{i=1}^{n} CU_i * MEM\ [classification] \quad (11)$$

where, SO indicates storage overhead, CU_i denotes the number of cloud users and MEM denotes the memory space consumed for classification. It is measured in terms of Megabytes (MB).

Figure 4 compares the classification accuracy of three different methods; namely, ORSDDSPNL-SDC, existing HBRSS [1], TCFDL [2] and Deep learning and IoT-based data processing frameworks [3] to several cloud users. Based on the classification, user authentication is performed for secure data access from the server. This significant enhancement of the ORSDDSPNL-SDC is verified through statistical estimation. As illustrated in Figure 4, increasing the number of cloud users causes a slow decrease in classification accuracy. This is because by increasing the number of cloud users, cloud users being classified accurately for secure data communication also gets deteriorated. This in turn causes a small decreasing trend in classification accuracy also. In the first iteration, we'll use 10000 cloud users to run our experiments. The accuracy of classifying authorized or unauthorized users using ORSDDSPNL-SDC is obtained as 99.65%. On the other hand, the accuracy of identifying authorized or unauthorized users by applying [1] and [2] is 98.1% and 98.8%, respectively. Likewise, different performance results are observed for each method. Overall, the observed results show that using ORSDDSPNL-SDC improves classification accuracy by 6% when compared to [1], 4% when compared to [2] and 5% when compared to [3].

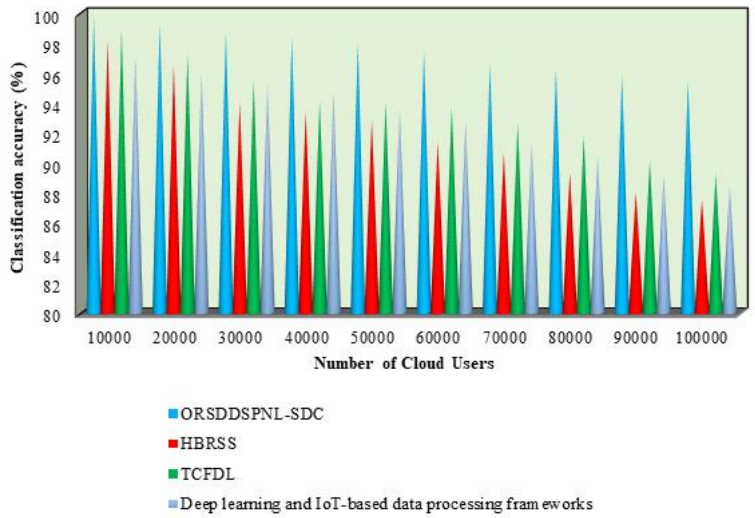


Figure 4. Comparison according to classification accuracy *versus* the number of cloud users.

Figure 4 depicts a comparison of classification accuracy for 100000 different cloud users. The figure shows that the number of cloud users is input in the x-axis and the overall performance results of classification accuracy are received in the y axis. According to Figure 4, the proposed ORSDDSPNL-SDC technique provides superior performance of classification accuracy as compared to HBRSS [1], and TCFDL [2] and deep learning and IoT-based data processing frameworks [3]. The existing TCFDL was employed to investigate security. But, maximum accuracy is achieved as a difficult issue in AI-enabled IoT. Contrary to existing works, orthogonal regressed steepest descent multilayer deep perceptive neural learning is utilized to enhance accuracy. This is owing to the cloud server authenticating the user ID with the already stored ID of that particular user at the time of registration. The verification is performed using the orthogonal regressed steepest descent multilayer deep perceptive neural learning. The orthogonal regression is applied to find the authorized users by using Szymkiewicz–Simpson coefficient. The Szymkiewicz–Simpson coefficient matches the user ID. The max-out activation function analyzes the similarity coefficient values and returns the classification result. Finally, the authorized user or unauthorized user classification results are getting more accurate at the output layer than in the existing methods.

The experimental results of the error rate for various categorization outcomes using three methods; namely, ORSDDSPNL-SDC, HBRSS [1] and TCFDL [2] and deep learning and IoT-based data processing frameworks [3] are shown in Figure 5. The observed results indicate that the overall performance of ORSDDSPNL-SDC technique is found to be minimized when compared to the other existing approaches; HBRSS [1] and TCFDL [2] and deep learning and IoT-based data processing frameworks [3]. This is proved through statistical examination. Let us consider 10000 users for calculating the error rate. By applying the ORSDDSPNL-SDC technique, the observed error rate is 0.35%. By considering [1], [2] and deep learning and IoT-based data processing frameworks [3], the observed error rates are 1.9%, 1.2% and 1.2%, respectively. Totally ten diverse results are observed for each method. The observed results of the ORSDDSPNL-SDC technique are compared to the existing methods. When compared to the existing methods, the obtained comparison result shows that the average error rate using the ORSDDSPNL-SDC technique is reduced by 71%, 64% and 66%, respectively. Figure 5 depicts the graphical representation of the ORSDDSPNL-SDC technique as well as of the three methods.

Figure 5 presents the result comparison of error rate versus the number of cloud users using different methods. However, the experiment is carried out with the number of 100000 cloud users to compute the error rate during the classification. From this result, the error rate using ORSDDSPNL-SDC technique was said to be reduced upon comparison with [1], [2] and deep learning and IoT-based data processing frameworks [3]. The reason behind the minimum error rate using ORSDDSPNL-SDC was owing to the application of the steepest descent function. After classification, the steepest descent function is applied for updating the weight between the layers of the multilayer deep perceptive neural learning classifier. As a result, the error rate of cloud user classification is minimized.

"Orthogonal Regressed Steepest Descent Deep Perceptive Neural Learning for IoT-aware Secured Big Data Communication", S. L. Swapna and V. Saravanan.

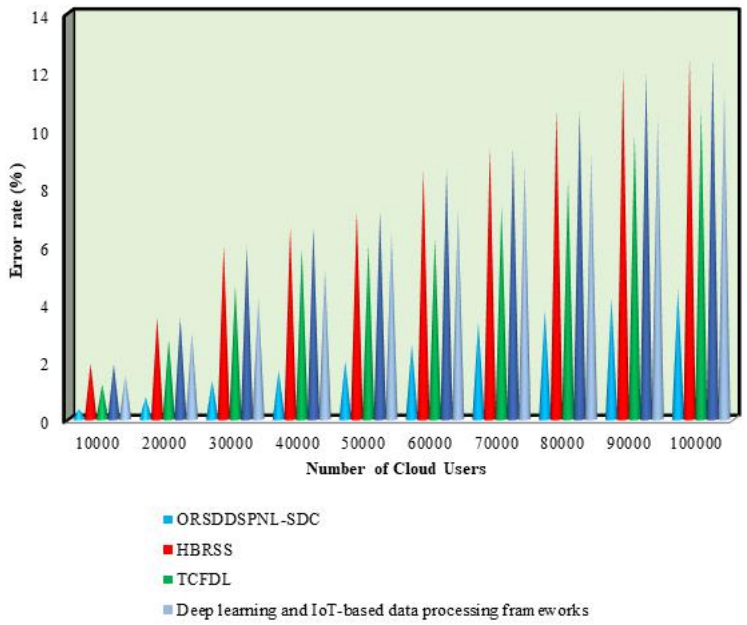


Figure 5. Comparison according to error rate *versus* number of cloud users.

Figure 6 illustrates the classification time using different methods; namely, ORSDDSPNL-SDC, HBRSS [1], TCFDL [2] and deep learning and IoT-based data processing frameworks [3]. From the observed results, the ORSDDSPNL-SDC technique decreases the classification time when compared to existing techniques. For example, with 10000 cloud users, the classification time using the ORSDDSPNL-SDC was observed at $3960ms$, whereas the classification times using [1] and [2] are $6100ms$ and $5500ms$. Similarly, different performance outcomes are obtained with each technique. The overall observed ORSDDSPNL-SDC results are compared to the results of existing methods. Ultimately, the mean of compared data reveals that the classification time using ORSDDSPNL-SDC technique is increased by 31% when compared to [1], 23% when compared to [2] and 13% when compared to [3].

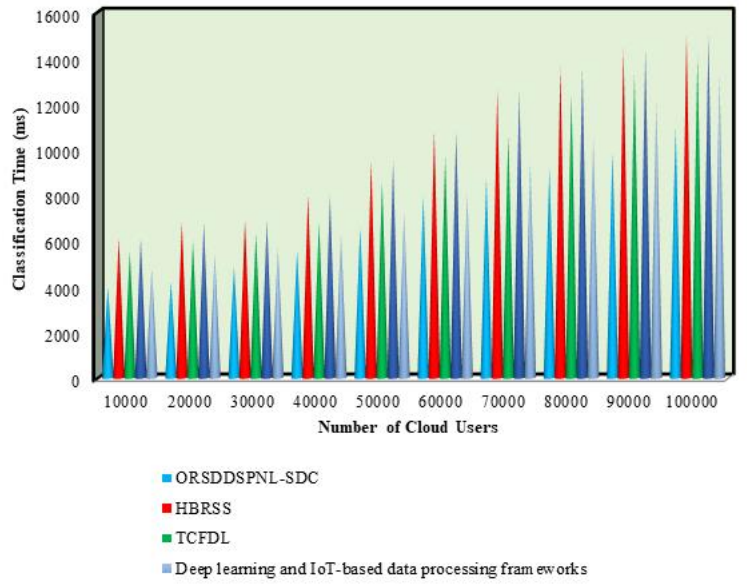


Figure 6. Comparison according to classification time *versus* the number of cloud users.

Figure 6 reveals the graphical representation of classification time for the different techniques for 100000 users. The graphical illustration implies that an increase in cloud users will lengthen the time required for classification. According to the evaluation's data, using the ORSDDSPNL-SDC reduces classification time compared to the existing [1], [2] and deep learning and IoT-based data processing frameworks [3], respectively. In traditional IoT systems, sensed data was often uploaded to the cloud that was examined with CFD. But, the data transmission causes a high classification time. The reason

behind this improvement is to perform the cloud user registration. In the ORSDDSPNL-SDC technique, the cloud users first register their details to the server for storing the data collected from the IoT device. After the registration, the cloud server generates the user ID and password for further processing. When the users access their data, the server identifies the users as authorized or unauthorized for secure data communication. Initially, the users' details are registered and then the server collects the data from the registered user. Then, classification is performed using deep learning with the help of the Szymkiewicz–Simpson coefficient by matching the ID of the user. Then, the max-out activation function returns the authorized and unauthorized user classification results. This process minimizes the time consumption of cloud user classification.

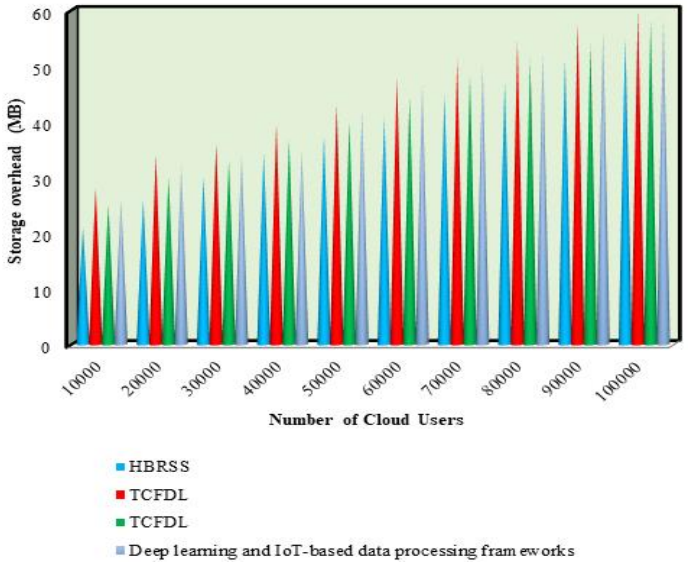


Figure 7. Comparison according to storage overhead *versus* the number of cloud users.

The performance analysis of the storage overhead using the different methods ORSDDSPNL-SDC, HBRSS [1], TCFDL [2] and deep learning and IoT-based data processing frameworks [3] is shown in Figure 7. The observed results indicate that the performance of memory consumption of the ORSDDSPNL-SDC technique is relatively minimum compared to the existing methods. With the consideration of 10000 users, the memory consumption to perform classification was found to be 21 MB. However, the memory consumption of existing [1] and [2] was found to be 28$MB$and 25 MB. The observed results indicate that the ORSDDSPNL-SDC technique reduces memory consumption. The overall results of memory consumption are compared to the previous findings after getting the ten results. In comparison to [1], [2] and [3], the comparison analysis shows a 15%,17% and 11% reduction in the approved user classification's memory use, respectively. Cloud storage, as well as cloud services, offer stronger computing power as well as distributed computing ability for IoT users at a lesser cost. But, the security problems of the cloud limit the growth of cloud computing and storage. The reason behind reducing memory consumption is applying orthogonal regressed steepest descent multilayer deep perceptive neural classifier. The proposed classifier accurately classifies the user as authorized or unauthorized with lesser memory consumption.

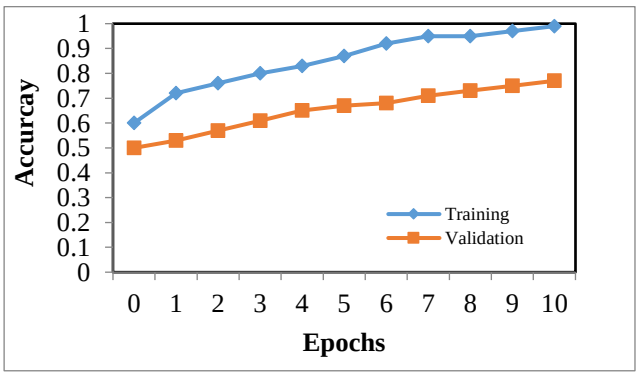


Figure 8. Training and testing accuracy *vs.* epochs.

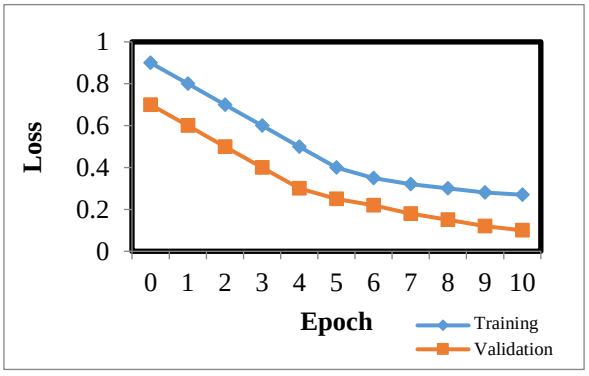


Figure 9. Training and testing loss *vs.* epochs.

In classification, data is used for divided into two parts; one for training and the other for testing. The training's accuracy will be higher than the testing's accuracy. In addition, the training's loss will be smaller than the testing's loss. The accuracy and loss are estimated in terms of percentage (%).

Figure 8 and Figure 9 show the impact of accuracy and loss along with the epoch using the Mobile HEALTH dataset. An epoch is taken in the horizontal direction, while accuracy and loss are observed at the vertical axis. As shown in the graphical chart, there are two various colors of lines such that blue and brown indicating the two parts; namely, training and validation, respectively.

6. CONCLUSION

While the Internet of Things (IoT) strengthens the viability of consumers in cloud environments, a lack of security practices raises the potential risks of protecting sensitive user data. One essential capability of big data communication is the use of IoT devices to secure data transmission. Therefore, a novel technique called ORSDDSPNL-SDC is introduced for guaranteeing secure communication by identifying the authorized cloud users to access the services from the cloud. The cloud users register their details to the server for accessing the various services from the cloud. The server creates a new ID and password for each user in the cloud. The data is then sent to the cloud server for storage by the cloud user. After that, the orthogonal regressed steepest descent multilayer deep perceptive neural learning is employed in ORSDDSPNL-SDC to analyze the receiver ID with the registered ID based on Szymkiewicz–Simpson coefficient. The Maxout activation function correctly classifies the user as authorized or unauthorized based on the similarity coefficient result. To minimize the error, the steepest descent function is then employed, increasing the classification accuracy of the users. In this way, ORSDDSPNL-SDC performs secured communication with authorized cloud users. The comprehensive experimental assessment is carried out using different performance metrics, such as classification accuracy, error rate, classification time and storage overhead *versus* the number of cloud users. The quantitative analysis confirms that the ORSDDSPNL-SDC technique has achieved higher accuracy of user classification with lesser time consumption as well as storage overhead when compared to other conventional methods.

REFERENCES

[1] H. Xie, Z. Zhang, Q. Zhang, S. Wei and C. Hu, "HBRSS: Providing High-secure Data Communication and Manipulation in Insecure Cloud Environments," Computer Comm., vol. 174, pp. 1-12, 2021.

[2] Q. Zhang, H. Zhong, W. Shi and L. Liu, "A Trusted and Collaborative Framework for Deep Learning in IoT," Computer Networks, vol. 193, pp. 1-10, DOI: 10.1016/j.comnet.2021.108055, 2021.

[3] Y. Li, Y. Zuo, H. Song and Z. Lv, "Deep Learning in Security of Internet of Things," IEEE Internet of Things Journal, vol. 9, no. 22, pp. 22133–22146, 2022.

[4] S. Zargar, A. Shahidinejad and M.Ghobaei-Arani, "A Lightweight Authentication Protocol for IoT-based Cloud Environment," Int. J. of Communication System, vol. 34, no.11, pp. 1-17, 2021.

[5] J. Shen, D. Liu, Q. Liu, X. Sun and Y. Zhang, "Secure Authentication in Cloud Big Data with Hierarchical Attribute Authorization Structure," IEEE Transactions on Big Data, vol. 7, no. 4, pp. 668 – 677, 2021.

[6] U. Iqbal, A. Tandon, S. Gupta, A. R. Yadav, R. Neware and F. W. Gelana, "A Novel Secure Authentication Protocol for IoT and Cloud Servers," Wireless Communications and Mobile Computing, vol. 2022, pp. 1-17, DOI: 10.1155/2022/7707543, 2022.

[7] A. Karati, R. Amin, S. K. H. Islam and K. R. Choo, "Provably Secure and Lightweight Identity-based Authenticated Data Sharing Protocol for Cyber-physical Cloud Environment," IEEE Transactions on Cloud Computing, vol. 9, no. 1, pp. 318 – 330, 2021.

[8] M. Wazid, J. Singh, A. K. Das, S. Shetty, M. K. Khan and J. J. P. C. Rodrigues, "ASCP-IoMT: AI-enabled Lightweight Secure Communication Protocol for Internet of Medical Things," IEEE Access, vol. 10, pp. 57990 – 58004, 2022.

[9] M. A. Jan, F. Khan, S. Mastorakis, M. Adil, A. Akbar and N.Stergiou, "LightIoT: Lightweight and Secure Communication for Energy-efficient IoT in Health Informatics," IEEE Transactions on Green Communications and Networking, vol. 5, no. 3, pp. 1202 – 1211, 2021.

[10] X. Lu, L. Yin, C. Li, C. Wang, F. Fang, C. Zhu and Z. Tian, "A Lightweight Privacy-preserving Communication Protocol for Heterogeneous IoT Environment," IEEE Access, vol. 8, pp. 67192–67204, DOI: 10.1109/ACCESS.2020.2978525, 2020.

[11] Z. Guan, W. Yang, L. Zhu, L. Wu and R. Wang, "Achieving Adaptively Secure Data Access Control with Privacy Protection for Lightweight IoT Devices," Science China Information Sciences, vol. 64, pp. 1-14, DOI: 10.1007/s11432-020-2957-5, 2021.

[12] F. M. Awaysheh, M. N. Aladwan, M. Alazab , S. Alawadi, J. C. Cabaleiro and T. F. Pena, "Security by Design for Big Data Frameworks Over Cloud Computing," IEEE Transactions on Engineering Management, vol. 69, no. 6, pp. 3676–3693, 2022.
[13] L. Atiewi, A. Al-Rahayfe, M. Almiani, S. Yussof, O. Alfandi, A. Abugabah and Y. Jararweh, "Scalable and Secure Big Data IoT System Based on Multifactor Authentication and Lightweight Cryptography," IEEE Access , vol. 8, pp. 113498 – 113511, DOI: 10.1109/ACCESS.2020.3002815, 2020.
[14] K. Thilagam, A. Beno, M. V. Lakshmi et al., "Secure IoT Healthcare Architecture with Deep Learning-based Access Control System," Journal of Nanomaterials, vol. 2022, pp. 1-8, DOI: 10.1155/2022/2638613, 2022.
[15] R. Li, H. Asaeda and J. Wu, "DCAuth: Data-centric Authentication for Secure In-network Big-data Retrieval," IEEE Transactions on Network Science and Engineering, vol. 7, no. 1, pp.15 – 27, 2020.
[16] U. Narayanan, V. Paul and S. Joseph, "A Novel System Architecture for Secure Authentication and Data Sharing in Cloud Enabled Big Data Environment," J. of King Saud Uni.-Computer and Information Sciences, vol. 34, no. 6, pp. 3121-3135, DOI: 10.1016/j.jksuci.2020.05.005, 2022.
[17] O. Kwabena, Z. Qin and T. Zhuang "MSCryptoNet: Multi-scheme Privacy-preserving Deep Learning in Cloud Computing," IEEE Access, vol. 7, pp. 29344–29354, DOI: 10.1109/access.2019.2901219, 2019.
[18] M. I. Ahmed and G. Kannan, "Secure End to End Communications and Data Analytics in IoT Integrated Application Using IBM Watson IoT Platform," Wireless Personal Communications, vol. 120, pp.153–168, DOI: 10.1007/s11277-021-08439-7, 2021.
[19] T. Maitra, S. Singh, R. Saurabh and D. Giri, "Analysis and Enhancement of Secure Three-factor User Authentication Using Chebyshev Chaotic Map," J. of Inf. Security and Appli., vol.61, pp.1-13, 2021.
[20] J. Sun, H. Xiong, X. Liu, Y. Zhang, X. Nie and R. H. Deng, "Lightweight and Privacy-aware Fine-grained Access Control for IoT-oriented Smart Health," IEEE IoT J., vol. 7, no. 7, pp. 6566- 6575, 2020.
[21] L. Zhang, Y. Shi, Y. Chang and C. Lin, "Hierarchical Fuzzy Neural Networks with Privacy Preservation for Heterogeneous Big Data," IEEE Transactions on Fuzzy Systems, vol. 29, no. 1, pp. 46–58, 2021.
[22] S. Banerjee, S. Roy, V. Odelu et al., "Multi-authority CP-ABE-based User Access Control Scheme with Constant-size Key and Ciphertext for IoT Deployment," J. of Information Security and Applications, vol. 53, pp. 1-22, DOI: 10.1016/j.jisa.2020.102503, 2020.
[23] S. Xu, Y. Li, R. H. Deng, Y. Zhang, X. Luo and X. Liu "Lightweight and Expressive Fine-grained Access Control for Healthcare Internet-of-Things," IEEE Transactions on Cloud Computing, vol. 10, no. 1, pp. 474–490, 2022.
[24] P. Abirami and S. V.Bhanu, "Enhancing Cloud Security Using Crypto-deep Neural Network for Privacy Preservation in Trusted Environment," Soft Computing, vol. 24, pp. 18927–18936, DOI: 10.1007/s00500-020-05122-0, 2020.
[25] S.Atiewi, A. Al-Rahayfeh, MuderAlmiani et al., "Scalable and Secure Big Data IoT System Based on Multifactor Authentication and Lightweight Cryptography," IEEE Access, vol. 8, pp. 113498–113511, DOI: 10.1109/ACCESS.2020.3002815, 2020.

ملخص البحث:

مع ازدياد انتشار أجهزة إنترنت الأشياء، يتمّ توليد كمياتٍ هائلةٍ من البيانات كُلَّ يوم. وتحتاج البيانات التي يتمّ جمعها الى تحديثٍ مستمرٍ في جهاز الخادِم السّحابي. إلى جانب ذلك، فإنّ نقل البيانات في بيئة السَّحابة بواسطة الإنترنت يواجه العديد من التّهديدات. ومع ذلك فإن قضية الأمْن الخاصّ بالبيانات تنقصها الفعالية في مجال الاتّصال بالبيانات الضّخمة. لذا، فإنّ هذه الورقة تهدف الى اقتراح تقنية مبتكرة للحفاظ على أمْن البيانات الضّخمة؛ إذ تتكون التّقنية المقترحة من ثلاث مراحل هي: التّسجيل، والتّحقُّق من هوية المستخدم، والنّقل الآمن للبيانات. وتتميز التّقنية المقترحة بدقّةٍ أعلى وزمنٍ أقل مقارنة بالتّقنيات التي تمّ استخدامها في دراساتٍ سابقة. حيث يقوم المستخدم بإدخال رمزٍ تعريفي ويقوم النّظام بمقارنته برقمٍ تعريفي مخزّن في جهاز الخادِم، مما يؤدّي الى تصنيف المستخدمين الى مستخدمين مخوّلين وآخرين غير مخوَّلين، بناءً على عددٍ من العوامل، ومنها: دقة التّصنيف، والزمن المستغرَق في التّصنيف، ومعدَّل الخطأ. وقد أثبت التّقييم نجاعة التّقنية المستخدَمة في هذا البحث من حيث تحقيق دقّة تصنيفٍ أعلى وزمن تصنيف أقلّ ومعدّل خطأ أقلّ، مقارنةً بالطّرق القائمة التي استُخدمت في دراسات أخرى.

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) مجلة علمية عالمية متخصصة محكمة تنشر الأوراق البحثية الأصيلة عالية المستوى في جميع الجوانب والتقنيات المتعلقة بمجالات تكنولوجيا وهندسة الحاسوب والاتصالات وتكنولوجيا المعلومات.

تحتضن وتنشر جامعة الأميرة سمية للتكنولوجيا (PSUT) المجلة الأردنية للحاسوب وتكنولوجيا المعلومات، وهي تصدر بدعم من صندوق دعم البحث العلمي في الأردن. وللباحثين الحق في قراءة كامل نصوص الأوراق البحثية المنشورة في المجلة وطباعتها وتوزيعها والبحث عنها وتنزيلها وتصويرها والوصول إليها. وتسمح المجلة بالنسخ من الأوراق المنشورة، لكن مع الإشارة إلى المصدر.

الأهداف والمجال

تهدف المجلة الأردنية للحاسوب وتكنولوجيا المعلومات (JJCIT) إلى نشر آخر التطورات في شكل أوراق بحثية أصيلة وبحوث مراجعة في جميع المجالات المتعلقة بالاتصالات وهندسة الحاسوب وتكنولوجيا المعلومات وجعلها متاحة للباحثين في شتى أرجاء العالم. وتركز المجلة على موضوعات تشمل على سبيل المثال لا الحصر: هندسة الحاسوب وشبكات الاتصالات وعلوم الحاسوب ونظم المعلومات وتكنولوجيا المعلومات وتطبيقاتها.

الفهرسة

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات مفهرسة في كل من:

فريق دعم هيئة التحرير

عنوان المجلة

الموقع الإلكتروني: www.jjcit.org
البريد الإلكتروني: jjcit@psut.edu.jo
العنوان: جامعة الاميرة سمية للتكنولوجيا، شارع خليل السكاكت، الجبيهة، عمان، الأردن.
صندوق بريد: 1438 عمان 11941 الأردن
هاتف: +962-6-5359949
فاكس: +962-6-7295534

جامعة الأميرة سميّة للتكنولوجيا
Princess Sumaya University for Technology

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات

ISSN 2415 - 1076 (Online)
ISSN 2413 - 9351 (Print)

آذار ٢٠٢٣ | المجلد ٩ | العدد ١

JJCIT

الصفحات	عنوان البحث
١ - ١٠	نموذج للكشْف المبكّر عن الهجمات التي تهدف الى كسْر كلمة السّرّ للوصول الى البيانات ووسْم مجموعات البيانات رماح يونس، محمد الكساسبة، محمد المسيعدين، و حمزة عبدي
١١ - ٢٠	هَوائي ذو رقْعةٍ ثلاثي النّطاقات باستخدام بِنية أساس مشوّهةٍ تتمّ أمثَلَتُهُ بواسطة خوارزمية جينية للتّطبيقات اللّاسلكية المتنقلة الحديثة خديجة أبو حسوس، ليلى واكريم، اسماء زوغاري، و عالية زكريتي
٢١ - ٣٥	فاعلية نماذج صِفرية الرّمية في التّوليد الأوتوماتيكي للشِّعر العربي محمد الغالي بِهيت، و مُعزّ بن حاج حميدة
٣٦ - ٥٢	تفسير وثاقة الصِّلة بين السِّمات المميّزة للنّصوص ومقروئيتها في نماذج توقّع المقروئية صفاء بريشي، نوال ناصري، عز الدين مزروعي، و عبد الحق لاخوجا
٥٣ - ٦٢	نموذج للكشْف عن خِطاب الكراهية على منصّات التّواصل الاجتماعي كينِدي مالِنغا ندنغا
٦٣ - ٧٥	نظام مقترح لضمان أمن البيانات عند انتقالها في إنترنت الأشياء عبر فَرْز المستخدِمين س. ل. سوابنا، و ف. سارافانان

www.jjcit.org

jjcit@psut.edu.jo

مجلة علمية عالمية متخصصة تصدر
بدعم من صندوق دعم البحث العلمي والابتكار