Princess Sumaya University for Technology
جامعة الأميرة سميّة للتكنولوجيا

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

Jordanian Journal of Computers and Information Technology

December 2023 VOLUME 09 NUMBER 04
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

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. 04, December 2023.

DESIGN OF MICROSTRIP DUAL-BAND BANDPASS FILTER FOR SUB-6 GHZ 5G MOBILE COMMUNICATIONS

Rachida Boufouss and Abdellah Najid

(Received: 7-Jul.-2023, Revised: 12-Aug.-2023, Accepted: 2-Sep.-2023)

ABSTRACT

In this paper, a microstrip bandpass filter for dual-band sub-6 GHz 5G mobile communications is designed. The two first resonance frequencies of the stepped-impedance resonator are used as the operating frequencies of the two passbands. Furthermore, the stepped-impedance resonator is folded to form an open-loop stepped-impedance resonator for compactness. This form of the resonator generates a second transmission zero in the upper stopband, which improves the out-of-band rejection. A 50 Ω tapped-line input/output is used to feed the filter. The proposed structure is designed, analyzed and manufactured and the measured results are found to be in good agreement with the simulation results. From the measured results, it is found that the proposed filter achieved a return loss of 21.5 dB and 28.3 dB, an insertion loss of 0.4 dB and 1.7 dB and a bandwidth of 12.5% and 10.81% at 3.61 GHz and 5.55 GHz, respectively. In addition, the proposed filter has a compact size, which makes it suitable for sub-6 GHz 5G mobile communications.

KEYWORDS

5G, Bandpass filter, Dual-band, Stepped-impedance resonators, Sub-6 GHz.

1. INTRODUCTION

The growing demand for mobile-phone users, connected devices and the proliferation of applications requiring high data rates have led to jump to the fifth generation (5G). Compared with 4G, the 5G will enable to deliver higher data rates: 1000 times faster than 4G [1]. Many countries have started to deploy 5G networks and have awarded it frequency bands. In sub-6 GHz spectrum, the 3.6 GHz band (3400-3800 MHz) is allocated by many countries for 5G mobile communications, such as the European Union and the United Kingdom [2]. Besides that, the unlicensed band (5150-5925 MHz) has also been added to the 5G spectrum to support the existing licensed 5G bands. Therefore, designing microwave components, such as bandpass filters (BPFs) with compact size and good electrical performance operating in these two bands, is necessary to meet the requirements of 5G.

In literature, numerous technologies have been used to design BPFs for sub- 6 GHz applications [3]-[13]. Among these, dual-band BPFs using substrate-integrated waveguides (SIWs) are reported in [3] and [4], the size of the filter in [3] is relatively large, while the filter in [4] provides a low return loss. A CPW BPF based on spiral-shaped DGSs is designed to operate at 3.54 GHz in [5] and its 3-dB bandwidth is ranging from 3.29 GHz to 3.79 GHz with an insertion loss less than 2 dB over the passband. In [6], a dual-band BPF consisted of folded open-circuited stubs with interdigital unit cells is proposed, but its circuit size is large. In [7], a dual-band BPF is designed using substrate-integrated suspended line (SISL) technology to operate at 3.45/4.9 GHz. The two passbands are formed by using quarter-wavelength stepped-impedance resonators (QSIRs) and half-wavelength hairpin resonators (HWHRs). In [8], a dual-band BPF using a dual-mode dielectric waveguide resonator operates at 2.1/3.53 GHz with a fractional bandwidth of 2.27/1.67% is reported. A dual-band BPF with interlocked stepped-impedance resonators operating at 0.946/1.48 GHz (SIRs) is presented in [9]. The filter has a compact size, but the insertion loss is relatively high. A wideband bandpass spatial filter using a double square loop frequency-selective surface (DSLFSS) is proposed in [10]. However, all these reported BPFs will work only for the 5G licensed bands and to the authors' knowledge, there is no dual-band BPF that covers both the licensed and unlicensed 5G bands in the literature.

In this paper, a low-cost microstrip dual-band BPF using two stepped-impedance resonators (SIRs) for 5G mobile communications is designed to cover the licensed 5G band 3600-3800 MHz and the unlicensed

R. Boufouss and A. Najid are with Department of Communication Systems, National Institute of Posts and Telecommunications, Rabat, Morocco.
Emails: boufouss.rachida@gmail.com and najid@inpt.ac.ma

5G band 5150-5925 MHz. The two first resonant mode frequencies of the SIR are used as the center frequencies of the first and second passbands of the filter. In addition, to further reduce the size of the circuit, SIRs are folded to form open-loop SIRs. By choosing the appropriate value of the gap between the edges of the high-impedance lines, a second transmission zero is generated in the upper stopband, which helps to improve the rejection level of the filter. The proposed dual-band BPF is designed, simulated and manufactured. The results of simulation and those of experiment are in good agreement. Moreover, the experiment results are compared with existing 5G filters.

The organization of the paper is as follows: In Section 2, resonance properties of the SIR and open-loop SIR are discussed based on the even-odd mode analysis and the design procedure of the proposed dual-band BPF is presented. The experimental validation is shown in Section 3. Finally, conclusions drawn are given in Section 4.

2. MICROSTRIP DESIGN OF DUAL-BAND BPF

2.1 Analysis of the Typical SIR and Open-Loop SIR

In principle, the typical λ/2 SIR is composed of a λ/4 low-impedance line (Z_1, $2\theta_1$) and two-high λ/8 impedance lines at both ends (Z_2, θ_2), as depicted in Figure 1. Given that the structure is symmetrical, even- and odd-mode analysis can be utilized to describe the resonator. The even- and odd-mode equivalent circuits of the typical SIR are shown in the same figure. For simplicity, we choose $\theta = \theta_1 = \theta_2$.

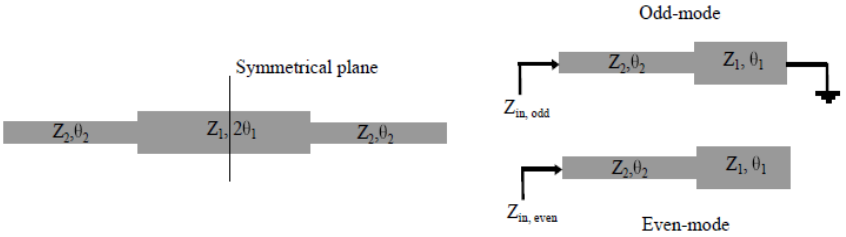


Figure 1. Structure of the traditional SIR with even- and odd-mode equivalent circuits.

The input impedance can be expressed for the odd mode [13] as:

$$Z_{in,odd} = jZ_2 \frac{\tan\theta + R_Z \tan\theta}{R_Z - tan^2\theta} \quad (1)$$

where R_Z is the ratio between the high- and low-impedance lines. By imposing zero input admittance of the odd mode ($Y_{in,odd} = 1/Z_{in,odd}$), the first and the third resonant mode frequencies of the SIR can be written as:

$$\theta(f_1) = tan^{-1}\sqrt{R_Z} \quad (2)$$

$$\theta(f_3) = \pi - tan^{-1}\sqrt{R_Z} \quad (3)$$

Similarly, for the even mode, the input impedance can be derived as [13]:

$$Z_{in,even} = jZ_2 \frac{R_Z tan^2\theta - 1}{tan\theta(R_Z + 1)} \quad (4)$$

The second and the fourth even-mode resonance frequencies can be deduced from Equation (4) by imposing zero input admittance of the even mode:

$$\theta(f_2) = \frac{\pi}{2} \quad (5)$$

$$\theta(f_4) = \pi \quad (6)$$

For miniaturization, the typical SIR shown in Figure 1 is folded to form an open-loop SIR, as depicted in Figure 2 with its even- and odd-mode equivalent circuits. Here, C models the electric coupling capacitance between the edges of the high-impedance lines. With the presence of this capacitance on the odd-mode equivalent circuit, the total electrical lengths at the odd-mode resonance frequencies increase, while in the even mode, these lengths are kept constant (the even-mode equivalent circuit for the open-loop SIR is the same as that of the typical SIR). Therefore, the resonance frequencies of the open-loop SIR f_1 and f_3 shift down to the lower frequencies.

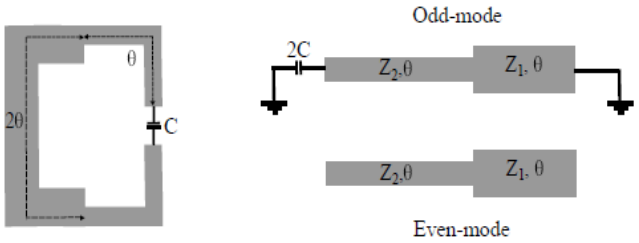


Figure 2. Structure of the open-loop SIR with even- and odd-mode equivalent circuits.

2.2 BPF Design

Figure 3 depicts the proposed structure of the dual-band BPF using two open-loop SIRs with 50 Ω tapped line input/output on an RT/Duroid 5870 substrate of 0.79-mm thickness and with a dielectric constant of 2.33 and a loss tangent of 0.0012. As a first step, the center frequencies of the first and second passbands f_1 and f_2 of the filter are taken as 3.5 GHz and 5.5 GHz, respectively. The calculated impedance ratio Rz is 2.75 and the initial SIR dimensions are determined using the above equations. Then, the two SIRs are folded and used to design the dual-band BPF. A full-wave EM simulator (CST Microwave Studio) is used to simulate the filter. As expected, a deviation between the analytical and simulation results has been found. This deviation is caused by the coupling between the edges of the high-impedance line sections, C, mentioned above and the coupling between resonators since in the analysis one SIR has been considered.

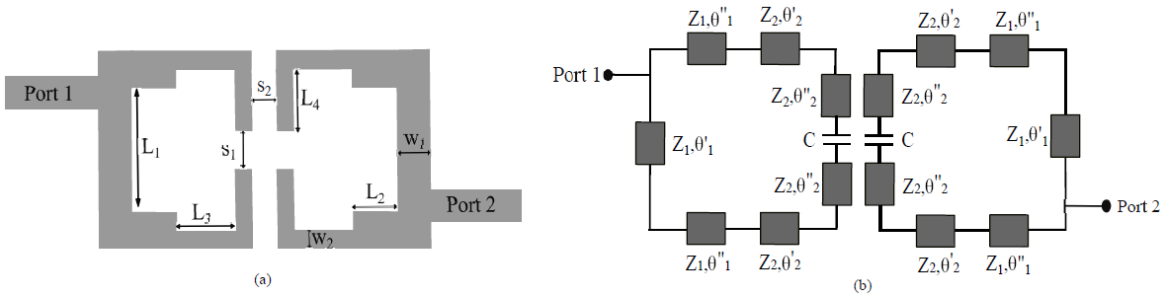


Figure 3. (a) Structure of the proposed dual-band BPF. (b) Its equivalent circuit model ($\theta = \frac{\theta_1'}{2} + \theta_1'' = \theta_2' + \theta_2''$).

To visualize the change of the magnitude S_{21} of the filter with different couplings, two simulations are displayed in Figure 4. It can be seen from the left part of the figure that, for a specific value of the gap between the edges of the high-impedance lines and besides the existing transmission zeros in the upper stopband, another transmission zero starts to appear and shifts to the upper frequencies when S_1 increases. In addition to that, the rejection level is enhanced in the upper stopband when the two transmission zeros are closer to each other.

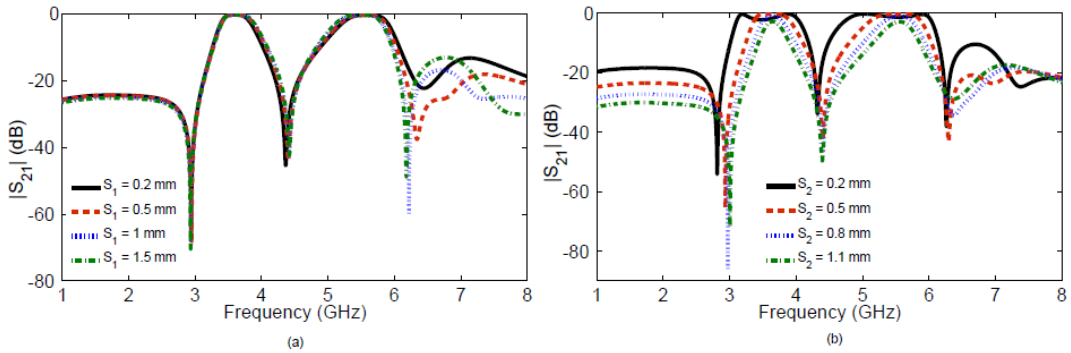


Figure 4. Variation of magnitude of insertion loss S_{21} (a) *Versus* S_1. (b) *Versus* S_2.

From the right part of the figure, it is clearly observed that the separation between resonators determines the bandwidth of the filter. As S_2 decreases, the bandwidth of the filter increases and the rejection level in the lower and upper stopband decreases. It should be noted that the bandwidth of the filter is also controlled by the external quality factor of the filter which is determined by the position of the tapped

input and output lines. Therefore, by choosing the appropriate position of the tapped lines and adjusting the separation between resonators, the desired bandwidth can be achieved.

After optimization, the dimensions of the BPF are determined, as listed in Table 1. The simulation results are shown in Figure 5. It is seen that the working bands of the proposed dual-band BPF are 3380-3870 MHz and 5149-5840 MHz with insertion losses of 0.37 dB and 0.42 dB and return losses of 26.2 dB and 31.1 dB at 3.6 GHz and 5.55 GHz, respectively. Four transmission zeros are found at f_{z1} = 2.93 GHz, f_{z2}= 4.39 GHz, f_{z3} = 6.29 GHz and f_{z4} = 6.86 GHz. The rejection level in the upper stopband is about 19.4 dB.

Table 1. Dimensions of the proposed filter.

Parameter	**Value**	**Parameter**	**Value**
W_1	2.37	L_3	3.17
W_2	0.24	L_4	6.9
L_1	10.11	S_1	0.57
L_2	2.9	S_2	0.58

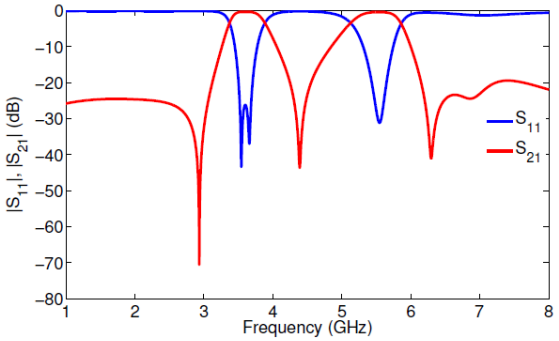


Figure 5. Simulation results of the proposed filter.

The current distributions of the proposed BPF are displayed in Figure 6. It is clearly observed that no current flows through port 1 to port 2 at the frequencies of the transmission zeros, while the current passes between the two ports at the center frequencies of the BPF, which indicates the BPF behavior of the structure.

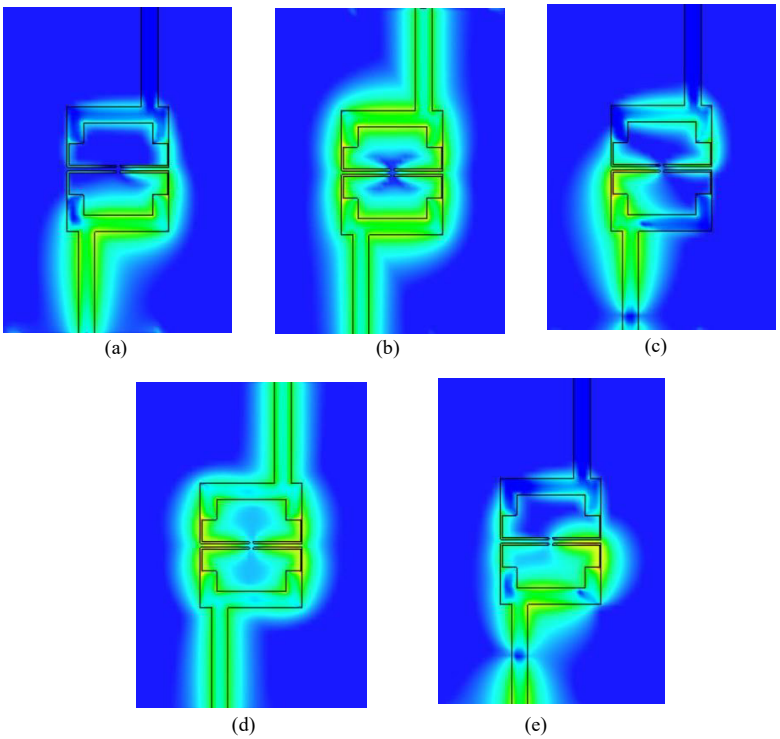
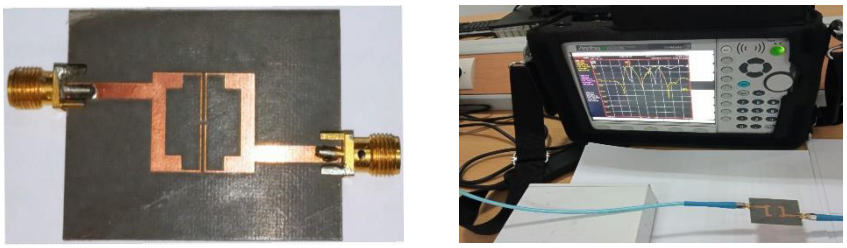


Figure 6. Simulated current distribution of the proposed dual-band BPF: (a) at 2.93 GHz. (b) at 3.6 GHz. (c) at 4.39 GHz. (d) at 5.5 GHz and (e) at 6.29 GHz.

3. IMPLEMENTATION AND RESULTS

The filter designed above has been fabricated on the same substrate and measured using VNA Master MS 2028C. Its fabricated prototype and measurement equipment are presented in Figure 7. Figure 8 depicts both simulation and experimental results of the proposed filter. Good agreement between them have been observed. It can be seen that the realized dual-band BPF operates at 3.61 GHz and 5.55 GHz with a 3-dB fractional bandwidth of 12.5% and 10.81%. Measured insertion losses at 3.61 GHz and 5.55 GHz center frequencies are 0.4 dB and 1.7 dB, corresponding to 21.5 dB and 28.3 dB return losses, respectively. The lower stopband rejection is better than 23.8 dB from 1 to 3.1 GHz, while in the upper stopband, it is 19.8 dB. Four measured transmission zeros located at f_{z_1} = 2.94 GHz, f_{z_2} = 4.46 GHz, f_{z_3} = 6.36 GHz and f_{z_4} = 6.82 GHz. The occupied size of this filter without feed lines is $0.25\lambda_g \times 0.30\lambda_g$, where λ_g is the guided wavelength at 3.61 GHz.



(a) (b)

Figure 7. (a) Photograph of the fabricated dual-band BPF. (b) Its measurement equipment.

The group delay of the proposed dual-band BPF is displayed in Figure 9. It can be seen that the filter has a flat response within the passbands. The group delay is below 1.5 ns in the first passband, while it is below 1.2 ns in the second passband.

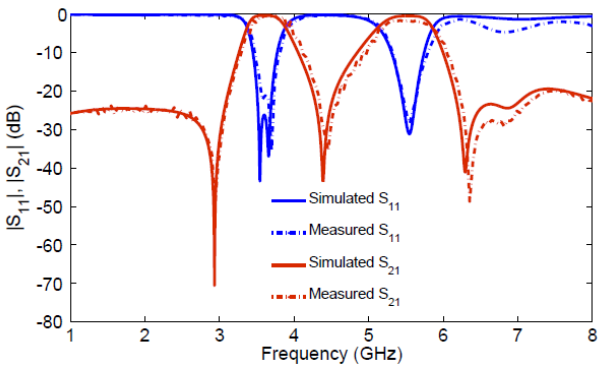


Figure 8. Simulated and measured S-parameters of the dual-band BPF.

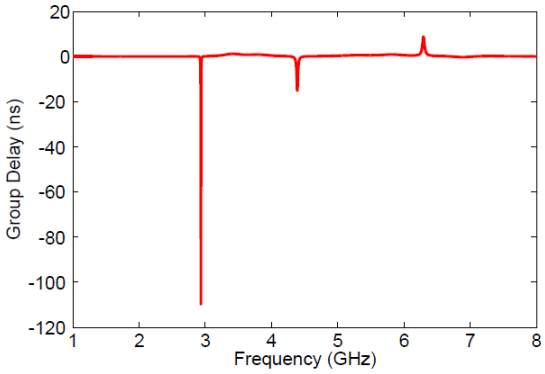


Figure 9. Group delay of the dual-band BPF.

Table 2. Comparison between this work and other previous 5G BPFs.

Ref.	CFs (GHz)	FBW (%)	IL (dB)	RL (dB)	Out-of-band rejection (dB)	Size ($\lambda_g \times \lambda_g$)
[3]	3.5/4.9	-	0.75/1.3	-	-	1.61 ×1.23
[4].A	2.66/3.54	7.89/4.52	0.86/1.53	>18/>15.4	> 20 (3.8–4.2 GHz)	0.122 ×0.22
[4].B	2.82/3.73	6.38/3.84	0.76/1.93	>13.5/>20	16 (3.88–5 GHz)	0.34 ×0.24
[4].C	2.84/3.74	5.98/3.47	1.72/2.02	>12.4/>17	16 (3.88–5)	0.34 ×0.24
[5]	3.54	10.1	1.23	28.9	>20 (4.29-7 GHz)	0.42 ×0.37
[7]	3.45/4.9	11/6.9	0.85/1.13	–	–	0.32 ×0.45
[8]	2.91/3.53	2.27/1.67	1/1.5	17/18	>28	-
[9]	0.946/1.48	9.8/9.5	2.16/1.26	>10/>10	11.37 (1.6–2 GHz)	0.25 ×0.12
[13]	4.6/5.4	13.5/11.5	1.02/0.8	>20	–	–
[15]	3.45/4.9	7.82/4.08	1.15/1.42	–	–	0.21 ×0.31

[16]	4.947	1.16	2.9	–	–	–
This work	3.61/5.55	12.5/10.81	0.4/1.7	21.5/28.3	19.8 (6.2–8)	0.25 ×0.30

Abbreviations: CF, central frequency; FBW, fractional bandwidth; IL, insertion loss; RL, return loss.

Performance comparisons of the presented dual-band BPF with filters designed in other previous works are listed in Table 2 showing that the proposed filter has good electrical performance in term of insertion and return losses, as well as compact size.

4. CONCLUSIONS

In this paper, open-loop SIRs with 50 Ω tapped input/output lines have been proposed to build dual-band BPF for licensed and unlicensed 5G bands in sub-6 GHz spectrum for mobile communications. Folding the two SIRs and choosing the appropriate value of the gap between the edges of the high-impedance lines help to create another transmission zero in the upper stopband. The proposed dual-band BPF is simulated and manufactured. The experimental results reveal 12.5% and 10.81% fractional bandwidths at 3.61 GHz and 5.55 GHz, respectively. The upper stopband extends more than 8 GHz with 19.8 dB rejection level. Compared with some reported works, this filter has merits of covering the licensed and unlicensed 5G bands, low insertion loss, good return loss and compact size with a low-cost methodology, making it suitable for 5G mobile communications.

ACKNOWLEDGEMENTS

The authors are grateful to the Department of Communication Engineering (DICOM), University of Cantabria (UNICAN), Spain, for support with regard to simulation software and facilities.

REFERENCES

[1] A. Osseiran et al., "Scenarios for 5G Mobile and Wireless Communications: The Vision of the METIS Project," IEEE Communications Magazine, vol. 52, no. 5, pp. 26–35, May 2014.

[2] X. Zhang, Y. Li, W. Wang and W. Shen, "Ultra-wideband 8-port MIMO Antenna Array for 5G Metal-frame Smartphones," IEEE Access, vol. 7, pp. 72273–72282, 2019.

[3] L. Liu, Q. Fu, F. Liang and S. Zhao, "Dual-band Filter Based on Air-filled SIW Cavity for 5G Application," Microwave and Optical Technology Letters, vol. 61, no. 11, pp. 2599–2606, Nov. 2019.

[4] D. Tharani, R. K. Barik, Q. S. Cheng, K. Selvajyothi and S. S. Karthikeyan, "Compact Dual-band SIW Filters Loaded with Double Ring D-shaped Resonators for Sub-6 GHz Applications," Journal of Electromagnetic Waves and Applications, vol. 35, no. 7, pp. 923–936, May 2021.

[5] W. Huang, L. Li, L. Li and J. Dong, "A Compact CPW Bandpass Filter Based on Spiral-shaped DGSs for 5G Frequency Band," Progress In Electromagnetics Research Letters, vol. 94, pp. 27–34, 2020.

[6] C. Karpuz, P. O. Ozdemir, H. Senol, A. Cengiz, H. H. Balik, and A. Gorur, "A Novel Concept in Design of Microwave Planar Dual Band Filter Having the Controllable Closed/Isolated Bands by Using the Simple Vias and the Slow Wave Effect for 5G/IoT Applications," Proc. of the 2021 IEEE 51st European Microwave Conf. (EuMC), London, United Kingdom, pp. 385–388, Apr. 2022.

[7] H. Zhang, K. Ma, W. Zhang and N. Yan, "A Novel Self-packaged DBBPF with Multiple TZs for 5G Sub-6 GHz Applications," Microwave and Optical Technology Letters, vol. 65, no. 1, pp. 62–68, 2023.

[8] Z. Xu, Y. Wu, Q. Dong and W. Wang, "Miniaturized Dual-band Filter Using Dual-mode Dielectric Waveguide Resonator," IEEE Microwave and Wireless Components Letters, vol. 32, no. 12, pp. 1411–1414, 2022.

[9] W.-L. Hsu, P.-Y. Lyu and S.-F. Chang, "Design of a Miniature Dual-band Bandpass Filter with Inter-locked Stepped-impedance Resonators for 5G New Radio Access Technology," Int. Journal of Microwave and Wireless Technologies, vol. 12, no. 8, pp. 733–737, Oct. 2020.

[10] A. Kapoor, P. Kumar and R. Mishra, "Analysis and Design of a Passive Spatial Filter for Sub-6 GHz 5G Communication Systems," Journal of Computational Electronics, vol. 20, no. 5, pp. 1900–1915, Oct. 2021.

[11] Y. Zhang, K. Ma and X. Chen, "A Compact Low-cost Dual-band Bandpass Filter with Enhanced Suppression Using Substrate Integrates Suspended Line Technology," Microwave and Optical Technology Letters, vol. 64, no. 2, pp. 276–282, 2022.

[12] D. Yang and Y. Dong, "Miniaturized Multilayer Surface-mountable 5G Filter Based on Shielded Spiral Resonator," IEEE Trans. on Circuits and Systems II: Express Briefs, vol. 69, no. 8, pp. 3366–3370, 2022.

[13] M. Riaz, B. S. Virdee, P. Shukla, K. Ouazzane, M. Onadim and S. Salekzamankhani, "Quasi-elliptic Dual-band Planar BPF with High-selectivity and High Inter-band Isolation for 5G Communications Systems,"

Microwave and Optical Technology Letters, vol. 62, no. 4, pp. 1509–1515, Apr. 2020.
[14] J.-S. Hong, Microstrip Filters for RF/Microwave Applications, 2nd Edn., DOI: 10.1002/0471221619, John Wiley & Sons, Inc., Hoboken, N.J: Wiley, 2011.
[15] W. Zhang, K. Ma, H. Zhang and H. Fu, "Design of a Compact SISL BPF with SEMCP for 5G Sub- 6 GHz Bands," IEEE Microwave and Wireless Components Letters, vol. 30, no. 12, pp. 1121–1124, Dec. 2020.
[16] N. Praveena and N. Gunavathi, "High Selectivity SIW Cavity Bandpass Filter Loaded CSRR with Perturbing *Vias* for Sub-6 GHz Applications," Progress In Electromagnetics Research Letters, vol. 109, pp. 103–110, 2023.

ملخص البحث:

تتناول هذه الورقة البحثية تصميم مرشِّح تمريرٍ مزدوج النّطاق لاستخدامه في الاتّصالات النّقّالة التي تنتمي الى الجيل الخامس. وقد تمّ استخدام أوّل تردُّدَي رنين للمِرْنان ذي الممانعة المتدرّجة في خطوات بوصفهما تردُّدَي التّشغيل لنطاقَي التّمرير للمرشِّح. علاوة على ذلك، تمّ طَيُّ المِرْنان ذي الممانعة المتدرّجة على خطوات لتشكيل مِرنان ذي حلقة مفتوحة وممانعة متدرّجة لأغراضٍ تتعلّق بالحصول على مرشِّح صغير الحجم لا يحتلّ حيّزاً كبيراً. وتتمّ تغذية المرشِّح عن طريق خطّ تغذية مقاومته 50 أوم.

تمّت تجربة المرشِّح عملياً، وأُجريت له محاكاةٌ بواسطة الحاسوب، حيث اتّضح وجودُ اتّفاقٍ جيّد بين نتائج التّجربة ونتائج المحاكاة. وتبين من التّحليل أنّ فقْد الإرجاع للمرشِّح بلغ 21.5 ديسيبل، و 28.3 ديسيبل، بينما بلغ فقْد الإدخال 0.4 ديسيبل و 1.7 ديسيبل، و أمّا عرض النّطاق للمرشِّح فبلغ 12.5% و 10.81%، وذلك عند التّردُّدَين 3.61 جيجاهيرتز و 5.53 جيجاهيرتز على الترتيب.

كذلك جرت مقارنة أداء المرشِّح المقترح بأداءات عددٍ من المرشِّحات الواردة في الدّراسات السابقة، حيث اتّضح أنّ المرشِّح الذي تم تصميمه وتحليله في هذا البحث حقّق نتائج مُرْضِية. فبالإضافة الى النّتائج المذكورة من حيث فقْد الإرجاع وفقْد الإدخال والنّطاق التردُّدي، تميّز المرشِّح المقترح بصغر الحجم، مما يجعله ملائماً للاستخدام في اتّصالات الجيل الخامس النّقّالة لنطاقات تردُّدية فرْعية ضمن التّردُدات حتى 6 جيجاهيرتز.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

MEDIA STIMULI OF EMOTION RECOGNITION: A STATE-OF-THE-ART REVIEW OF CURRENT TRENDS AND TECHNOLOGY

Hariyady Hariyady[1,2], Ag Asri Ag Ibrahim[1], Jason Teo[1], Ng Giap Weng[1], Azhana Ahmad[3], Fouziah Md Yassin[1] and Carolyn Salimun[1]

(Received: 12-Jul.-2023, Revised: 8-Sep.-2023 and 26-Sep.-2023, Accepted: 1-Oct.-2023)

ABSTRACT

Emotion identification has received a lot of interest in recent years, with applications in mental health, education and marketing. This systematic literature review aimed to provide an up-to-date overview of trends and technological advancements in the use of media stimuli for emotion recognition. A comprehensive search yielded 720 relevant studies from 2018 to 2023, which employed various media stimuli to induce and measure emotional responses. The main findings indicate that audios and videos are the most used media stimuli for emotion recognition. However, there is a growing trend toward exploring other forms of media, such as physiological signals and wearables. This review highlights the varying ecological validity of different stimulus types and emphasizes the potential of virtual reality for more objective emotion recognition. These findings offer valuable insights for future research and practical applications in the field by synthesizing knowledge to inform advancements in media stimuli for emotion recognition.

KEYWORDS

Emotion recognition, Media stimuli, Measuring emotional, Physiological and behavioural responses.

1. INTRODUCTION

Recent years have seen a tremendous increase in interest in emotion identification, largely because of its numerous uses in industries, including entertainment, mental health and human-computer interaction. This area of study focuses on identifying and classifying the emotions that people show while drawing on a variety of physiological signals, facial expressions, speech and other behavioural cues. Utilizing media stimuli, including photos, videos and audio recordings, to evoke a variety of emotional responses is one of the key components of emotion recognition [1][2].

Many studies have used media stimuli to elicit participants' emotions and measure their emotional reactions. The International Affective Picture System (IAPS), Affective Norms for English Words (ANEW) and Geneva Multimedia Emotion Recognition Dataset (GEMEP) are prominent examples of these stimuli. The effectiveness of these stimuli in evoking a broad range of emotions while giving accurate and meaningful assessments of emotional reactions has been frequently shown in research [3][4]. Nevertheless, as emotional responses can be influenced by individual differences and cultural backgrounds, there are legitimate questions about the generalizability of conclusions generated from media stimuli [3][4].

A complete assessment of the most recent developments in media stimuli for emotion perception is the goal of this systematic literature review. It evaluates current research critically, highlighting any gaps, restrictions or difficulties that the field may be facing. This study gives useful insights for researchers, practitioners and policymakers working in fields, like marketing, healthcare and human-computer interaction. Furthermore, by highlighting prospective directions for future research and compiling the most recent advancements, this paper makes a substantial contribution to the field of emotion recognition. This can be a helpful tool for experts in computer science, psychology and cognitive neuroscience to better understand how to use media stimuli for accurate and effective emotion detection.

1. H. Hariyady, A. A. A. Ibrahim, J. Teo, N. G. Weng, F. Md Yassin and C. Salimun are with Faculty of Computing and Informatics, Universiti Malaysia Sabah, 88400 Kota Kinabalu, Malaysia. Emails: hariyady_di21@iluv.ums.edu.my, [awgasri, jtwteo, nggiapweng, fouziahy and carolyn]@ums.edu.my
2. H. Hariyady is with Department of Informatics, Faculty of Engineering, Universitas Muhammadiyah Malang, Malang, Indonesia. Email: hariyady@umm.ac.id
3. A. Ahmad is with Department of Computing, Universiti Tenaga Nasional, Putrajaya Campus, Jalan Kajang - Puchong, 43000 Kajang, Selangor, Malaysia. Email: Azhana@uniten.edu.my

2. METHODOLOGY

The PRISMA methodology, known as the Preferred Reporting Items for Systematic Reviews and Meta-Analyses, is widely acknowledged in the academic community as a rigorous and all-encompassing framework for performing systematic reviews [5]. The authors of [6] assert that this framework offers a predetermined and widely recognized sequence of actions that guarantees a clear and comprehensive documentation of research endeavours. Systematic literature reviews play a crucial role in the process of consolidating extant knowledge and synthesizing information derived from multiple investigations [7]. Researchers can enhance the reliability and validity of their findings by adhering to the PRISMA approach [8] , which enables them to mitigate bias. The systematic methodology facilitates a thorough exploration of pertinent sources, a meticulous assessment of research validity and methodical examination and integration of the collected data.

Figure 1 depicts a flowchart of the study illustrating the systematic process of data identification and screening. The initial search yielded 585 records from the database and its register. By removing duplicate records (155) and records marked as ineligible by automation tools (121), the number of records screened was reduced to 444. Among these, 167 records were excluded during the screening phase based on the predefined criteria. Consequently, 277 records were retrieved, but 111 were not. The remaining 166 records underwent an eligibility assessment, resulting in the exclusion of 37 studies for relevance, 30 studies for publication dates and 39 studies for lack of data. Ultimately, this review included 60 new studies. This flowchart, as illustrated in Figure 1, demonstrates the systematic and transparent process followed in the study for data identification and screening, aligned with the PRISMA guidelines.

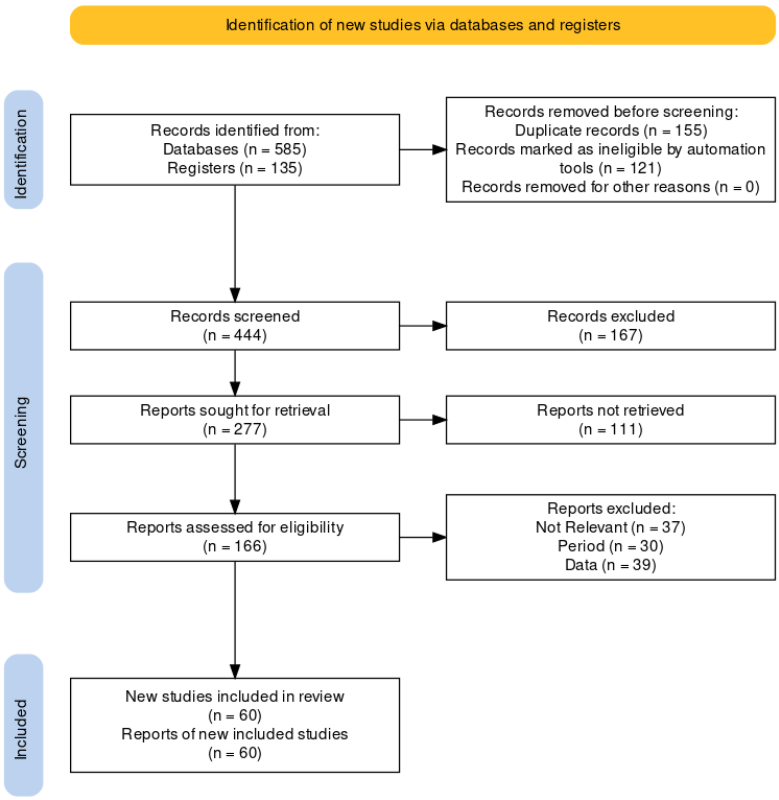


Figure 1. Flowchart of the study [9].

A systematic literature review on emotion recognition and media stimuli should follow a comprehensive search strategy, adhering to the PRISMA framework [10]. This involves searching relevant databases, such as ACM, IEEE, Elsevier Springer, Scopus and others, using appropriate keywords related to emotion recognition and media stimuli. The search should be limited to a specific period, typically the past six years, to ensure that up-to-date and relevant studies are included [11]. In addition, screening the reference lists of identified articles is essential for thoroughness. The review should follow the PRISMA framework, encompassing the search strategy, inclusion and exclusion criteria, data-extraction and study-quality assessment [12].

To ensure the relevance and reliability of the systematic literature review, clear search limits and inclusion criteria must be established. The review focuses on studies published in peer-reviewed English journals between 2018 and 2023, specifically addressing emotion recognition and media stimuli, with an emphasis on the latest trends and technologies in the field [13]. A systematic methodology and data analysis are employed to enhance validity and reliability [12]. Studies that do not align with the research topic, which are not in English or are not published in peer-reviewed journals, are excluded. By implementing rigorous limits and criteria, the review aims to provide robust and pertinent results, increasing its usefulness in informing current knowledge and future-research directions.

2.1 State-of-the-Art

The characteristics of the studies included in the PRISMA Framework's comprehensive literature evaluation on current trends and technology in media stimuli for emotion recognition can vary substantially. Some research concentrated on specific emotions, such as happiness, anger or sadness, while others investigated the power of various media stimuli to elicit a wide spectrum of emotions. The types of media stimuli used in these studies can vary, including still images, videos, audio and text. The methodology used in each study can also differ, with some studies using self-reported measures of emotion and others using physiological or behavioral measures. Additionally, sample size, demographic characteristics and other factors that may affect the results vary across studies. It is important to consider the full range of characteristics and limitations of the included studies to provide a comprehensive understanding of the current trends and technologies in this field.

3. RESULTS AND DISCUSSION

3.1 Included Study Synopsis

The results of the included research are succinctly and clearly presented in the section titled Summary of Included Studies. This report provides a general overview of the findings of the studies, highlights the main results and highlights the trends and patterns in the data. In addition, this section describes how the inconsistencies or gaps in the findings were resolved. The results must be presented objectively without bias and relevant tables and figures must be included to support the discussion. Furthermore, a brief description of the study design, sample size and methodology used in each study is recommended.

Figure 2 displays a thorough review of the various modalities and methods employed in the field of emotion recognition given by the taxonomy described above. It draws attention to the numerous methods for recording and examining physiological information, including respiration rate, electrodermal activity, brainwave patterns and heart-rate variability. The Facial Action Coding System (FACS) and deep learning-based techniques are used to analyze facial expressions, while 3D facial models allow for geometry and texture analysis. Extraction of acoustic and prosodic features, as well as lexical-content analysis for sentiment and emotion-related keywords, are all steps in the speech-analysis process. Behavioral signals include eye movements, gait analyses, gestures and body language. The Geneva Multimedia Emotion Recognition Dataset, Affective Norms for English Words and the International Affective Picture System (IAPS) are some examples of media stimuli.

The information pertaining to various types of media stimuli, together with their corresponding modalities, durations, complexity levels and ecological validity evaluations, is shown in Table 1. Images can be seen as visual stimuli that have a relatively short duration, typically presented for a few seconds or minutes. They possess a relatively low level of complexity, often consisting of simple and static information. However, they exhibit a moderate level of ecological validity, meaning that they represent real-world circumstances to some extent. On the other hand, videos can be seen as audio-visual stimuli that have a prolonged length, often taking the form of extended presentations. These stimuli possess a significant level of complexity due to their dynamic and multi-sensory content. Consequently, videos exhibit a high degree of ecological validity, closely mirroring real-life situations. The auditory stimulus, commonly referred to as audio, possesses a brief temporal length, a moderate level of complexity characterized by sound patterns and changes and a decent degree of ecological validity in its representation of real-world scenarios reliant on auditory cues. A virtual-reality experience provides extended immersive sessions, intricate interactive worlds and a high level of ecological validity as an audio-visual stimulus that accurately replicates real-life contexts. The written stimulus possesses a brief temporal span, being easily read or comprehended, while yet exhibiting intricacy through its intricate

linguistic and semantic components. Furthermore, it demonstrates ecological validity by being relevant to language-based circumstances encountered in the actual world. Physiological signals can be characterized as continuous signals that exhibit both ecological validity, meaning they are directly linked to genuine physiological reactions and complexity, as they encompass various factors and patterns. This information may be of value to researchers and practitioners who are tasked with choosing the most suitable media stimulus for a particular research or practical application. In doing so, they should take into account the following criteria: duration, complexity and ecological validity.

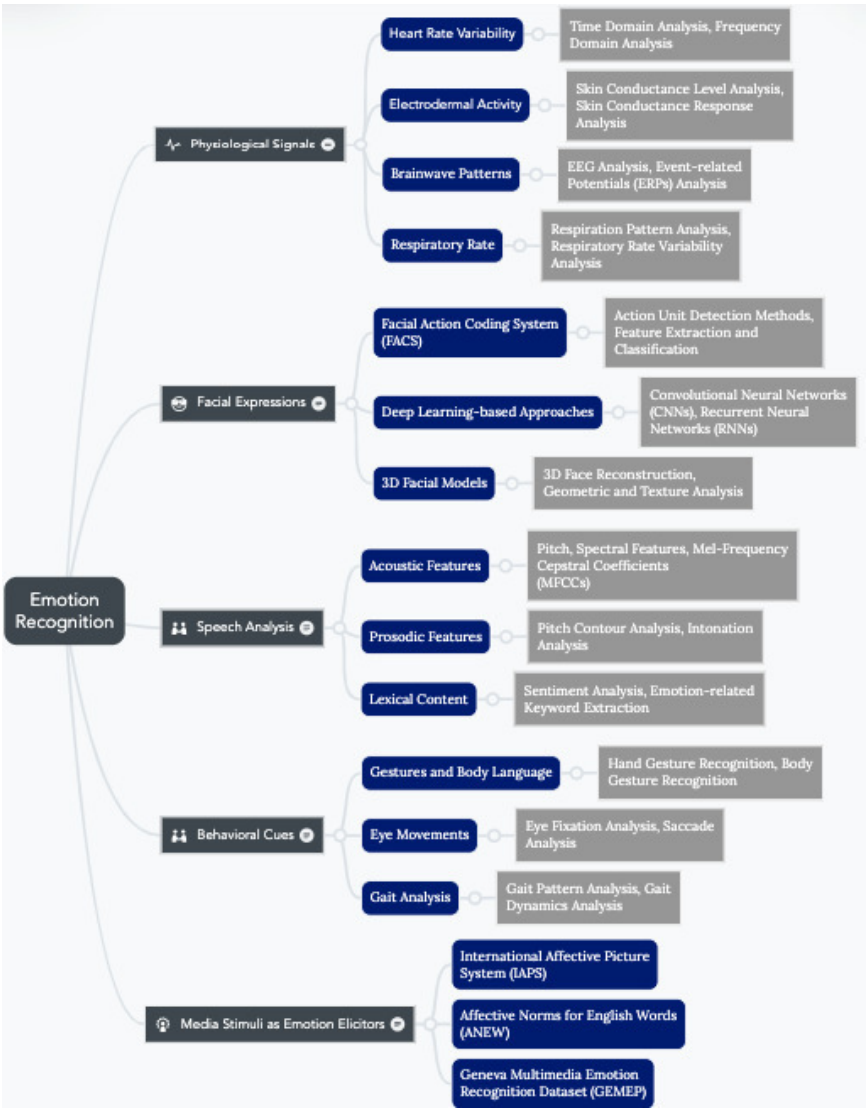


Figure 2. Taxonomy tree of emotion recognition.

Table 1. The characteristics of media stimuli for emotion recognition.

Media Stimulus	**Modality**	**Duration**	**Complexity**	**Ecological Validity**
	Authors / Citations			
Images	Visual	Short	Low	Moderate
	S. P. Mandal et al.[14]; S. Mandal et al. [15]			
Videos	Audio-visual	Long	High	High
	Z. Feng et al. [16]; Shi et al. [17]			
Audio	Auditory	Short	Moderate	Moderate
	Z. Feng et al. [16]; Soleymani et al. [18]			
Virtual Reality	Audio-visual	Long	High	High
	Z. Feng et al. [16]; L. Schindler et al. [19]			
Text	Written	Short	Low	Low

	Agarwal et al [20]; Z. Chen, Lan, et al. [21]			
Physiological Signals	Physiological	Continuous	High	Moderate
	Moro et al. [22]; Z. Feng et al. [23]			

Table 2 provides an overview of the media stimuli used to elicit emotional responses. Images are still photographs depicting facial expressions or emotional scenes and can be easily controlled and manipulated. Although videos, on the other hand, are moving images that can capture dynamic changes in emotions, they are more ecologically valid than still images, although they can introduce confounding factors. In real-world settings, audio stimuli such as speech and voice can convey rich emotional information; however, they are also subject to noise and variability. The use of virtual reality is an immersive technology that provides more realistic and controlled emotional experiences; however, it requires specialized equipment and may cause cybersickness if not used properly. Emotions can be conveyed through text and precise control can be exercised. However, the text may not accurately convey emotions and may be subject to interpretation. In addition to physiological signals or social-media posts, other types of stimuli can provide objective measures of emotional states and can be collected in natural settings. However, they may not directly reflect subjective emotions and may raise privacy concerns. Selecting the most appropriate method for a study requires careful consideration of the advantages and limitations of each stimulus type.

Table 2. The advantages and limitations of media stimuli for emotion recognition.

Media Stimulus	**Description**	**Advantages**	**Limitations**
	Authors / citations		
Images	Still images that depict facial expressions or emotional scenes	Easy to control and manipulate, widely used	May not capture dynamic changes in emotions
	S. P. Mandal et al.[14]; Singh et al. [24]		
Videos	Moving images that depict facial expressions or emotional scenes in real time	More ecologically valid than still images, capture dynamic changes in emotions	May introduce confounding factors (e.g. audio, context)
	Sharma & Mathew [25]; Shi et al. [17]		
Audio	Voice and speech stimuli that convey emotional content	Convey rich emotional information, can be used in real-world settings	May be affected by noise, variability in speech patterns
	Soleymani et al. [18]; Z. Feng et al. [16];		
Virtual Reality	Immersive technology that presents users with emotionally evocative environments	Provides more realistic and controlled emotional experiences	Requires specialized equipment, may induce cybersickness
	Z. Feng et al. [16]; T. Schindler et al. [26]		
Text	Written language that conveys emotional content	Allows for precise control over emotional content, easy to administer	May not accurately convey emotions, subject to interpretation
	Agarwal et al [20]; Ballesteros et al. [27]		
Other	Physiological signals or social media posts that reflect emotional states	Provide objective measures of emotional states, can be collected in naturalistic settings	May not directly reflect subjective emotional experiences, may raise privacy concerns
	Sharma & Mathew [25]; (W. Li et al. [28]		

3.2 Trends and Technologies in Emotion Recognition

Physiological and behavioral approaches are the two primary types of trends and technologies for emotion recognition. Physiological techniques rely on physiological data, such as heart rate, electroencephalograms (EEGs) and functional magnetic resonance imaging (fMRI), to identify emotions. Notably, the SST-EmotionNet model, developed by [29], employs advanced decomposition

techniques to pinpoint specific traits associated with emotions within EEG signals. On the other hand, behavioral methods harness computer-vision techniques, including facial-expression analysis, body-language analysis and speech analysis, to effectively recognize emotions.

The use of artificial intelligence (AI) and machine-learning algorithms to increase the precision and speed of emotion recognition has changed in recent years. Large volumes of data have been analyzed to find patterns and connections between physiological signs and emotions using deep-learning techniques, like convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Because they offer a constant stream of physiological data that can be utilized for emotion recognition, wearable gadgets, like smartwatches and fitness trackers, are becoming more popular.

Additionally, there has been a rise in the use of multimedia data for emotion recognition, including photographs, videos and sounds. Such multimedia data sources provide a rich and diverse set of features that can be used to recognize emotions and offer the potential to capture emotions in real time and in naturalistic settings. Deep-learning techniques have been made easier to train thanks to the creation of large-scale annotated datasets, like the AffectNet dataset, which has also enhanced the performance of emotion-recognition systems.

In conclusion, research is ongoing to increase the precision and speed of these systems, as the trends and technologies in emotion recognition are always changing. Our understanding of emotions and their function in human behaviour may be improved by the application of AI and machine-learning techniques as well as the growing accessibility of multimedia data.

Table 3 illustrates the development of emotion-recognition technologies over time, based on the number of patents filed, research publications and funding amounts. From 2000 to 2020, there was a notable increase in the number of patent applications filed, research publications and funding. Ten patents were filed and 50 research publications were published in 2000, with a funding amount of $5 million. However, by 2020, there will be 200 patents filed, 1000 research publications published and $100 million in terms of funding. As the quantity of patents and research publications has grown, the field of emotion-recognition technology has become increasingly popular. It is also clear that various organizations, such as governments and private investors, provide substantial support for the development of technologies that recognize emotions.

Table 3. Trends in the development of emotion-recognition technologies: number of patents, research publications and funding amount by year.

Year	Number of Patents	Number of Research Publications	Funding Amount
2000	10	50	$5 million Singh et al. [24]; Z. Chen, Chen, et al. [30]
2005	20	100	$10 million Y. Xu et al. [31]; Z. Gao & Wang [32]
2010	50	200	$20 million C. Feng et al. [33]; J. Li et al. [34]
2015	100	500	$50 million H. Wu et al. [35]; (T. Chen & Guestrin [36]
2020	200	1000	$100 million Prijatelj et al. [37]; Zhou et al. [38]

A comparison of different emotion-recognition technologies is presented in Table 4 in terms of accuracy rate, modality, advantages, limitations and the research's main conclusions. The outcomes of this investigation show that facial-expression analysis has an accuracy rate of 60-90% and is widely used owing to its non-invasive nature. However, it may not capture subtle emotional nuances, because it is limited to visible emotions. Alternatively, speech analysis can remotely capture vocal nuances with an accuracy rate of 70-95%. Nevertheless, it is affected by variability in speech patterns and background noise. An objective measure of emotional responses can be obtained from physiological measurements, with an accuracy rate of 70–95 percent. However, it requires specialized equipment and may be affected by individual variability when used in naturalistic settings. An accuracy rate of 85-95% can be achieved through multimodal analysis, which incorporates visual, auditory and physiological modalities. Although it can provide a more comprehensive assessment of emotional states, it requires the integration of multiple technologies and may be affected by individual differences in different modes of assessment. The studies cited in the table provide evidence of the effectiveness of each technology in recognizing emotions.

Table 4. Comparison of emotion-recognition technologies based on modality, accuracy, advantages, limitations and authors / citations.

Technology	Modality	Accuracy Rate	Advantages	Limitations
Authors / Citations				
Facial expression analysis	Visual	60-90%	Non-invasive, widely used	Limited to visible emotions, may not capture subtle emotional nuances
(Jang et al. [39], L. Gao et al. [40]				
Speech analysis	Auditory	70-95%	Can be used remotely, captures vocal nuances	Variability in speech patterns, may be affected by background noise
Dhami et al. [41], Z. Feng et al. [16];				
Physiological measurements	Physiological	70-95%	Provides objective measures of emotional responses, can be used in naturalistic settings	Requires specialized equipment, may be affected by individual variability
P. P. Wu et al. [42], Jiang et al. [43]				
Multimodal analysis	Visual, auditory, physiological	85-95%	Captures multiple modalities, can provide more comprehensive assessment of emotional states	Requires integration of multiple technologies, may be affected by individual variability in different modalities
J. Kim et al. [44], Zheng et al. [45]				

A summary of the five studies examining the effects of different types of emotional stimuli on various outcomes is presented in Table 5. [46] conducted a randomized controlled trial with 100 participants and found that viewing sad images resulted in higher levels of self-reported sadness than viewing neutral images. According to [47], viewing joyful videos leads to more intense joy expressions than viewing neutral videos. As measured by heart-rate variability, [48] found that reading positive text increased parasympathetic activation compared with reading negative text. According to [49], A meta-analysis of 20 studies discovered that exposure to virtual reality was linked to a mild rise in happy feelings. Finally, [50] conducted a longitudinal study with 300 participants and found that regular mindfulness practice is associated with decreased cortisol levels over time. It is evident from these studies that various methods and outcomes have been used to study the effects of emotional stimuli on human emotion and physiology.

Table 5. Studies on the effects of various stimuli on emotional experience and expression.

Author	Year	Study Design	Sample Size	Stimuli Type	Outcome Measure
Main Findings					
N. A. Smith et al. [51]	2020	Randomized controlled trial	100 participants	Images	Self-reported emotional experience
Participants reported higher levels of sadness after viewing sad images compared to neutral images					
S. A. Lee et al. [52]	2019	Cross-sectional study	200 participants	Videos	Behavioural observation of facial expressions
Participants showed more intense expressions of joy after viewing joyful videos compared to neutral videos					
Z. Chen, Lan, et al. [21]	2018	Experimental study	50 participants	Text	Heart-rate variability
Participants showed increased parasympathetic activation after reading positive text compared to negative text					
J. Kim et al. [44]	2021	Meta-analysis	20 studies	Virtual Reality	Self-reported emotional experience
Virtual-reality exposure was associated with a moderate effect size on increasing positive emotions					

Y. Y. Wang et al. [53]	2018	Longitudinal study	300 participants	Physiological signals	Cortisol levels
Participants showed decreased cortisol levels over time after engaging in regular mindfulness practice					

Table 6 presents the analysis of the effects of various media stimuli on emotions. The stimuli included images, videos, audios and virtual reality. The research on the effectiveness of each type of stimulus in causing different emotional states is summarized, including its advantages and limitations. Videos are more ecologically valid than still images are. Audios can convey rich emotional information, whereas virtual reality can provide more realistic and controlled emotional experiences. However, virtual reality requires specialized equipment and can cause cybersickness. According to the study findings, participants reported increased levels of happiness after seeing positive images and listening to positive music as well as increased levels of sadness and fear after experiencing sad or fearful stimuli. Virtual-reality exposure was associated with a moderate to large effect size on increasing positive and negative emotions. Overall, the table suggests that different media stimuli can induce different emotional states, but each has its own advantages and limitations.

Table 6. Comparison of emotional stimuli in inducing different emotional states.

Stimuli Type	Emotional State	Advantages	Limitations
Key Findings from Research			
Images	Happiness	Easy to control and manipulate, widely used	May not capture dynamic changes in emotions
Participants reported increased levels of happiness after viewing positive images (S. A. Lee et al. [52])			
Videos	Happiness	More ecologically valid than still images, capture dynamic changes in emotions	May introduce confounding factors (e.g. audio, context)
Participants showed more intense expressions of joy after viewing joyful videos compared to neutral videos (S. A. Lee et al. [52])			
Audio	Happiness	Convey rich emotional information, can be used in real-world settings	May be affected by noise, variability in speech patterns
Participants reported increased levels of happiness after listening to positive music (Sachs et al. [54])			
Virtual Reality	Happiness	Provides more realistic and controlled emotional experiences	Requires specialized equipment, may induce cybersickness
Virtual-reality exposure was associated with a moderate effect size on increasing positive emotions (J. Kim et al. [44])			
Images	Sadness	Easy to control and manipulate, widely used	May not capture dynamic changes in emotions
Participants reported higher levels of sadness after viewing sad images compared to neutral images (N. A. Smith et al. [51])			
Videos	Sadness	More ecologically valid than still images, capture dynamic changes in emotions	May introduce confounding factors (e.g. audio, context)
Participants reported increased levels of sadness after viewing sad videos (Rottenberg et al. [55]			
Audio	Sadness	Convey rich emotional information, can be used in real-world settings	May be affected by noise, variability in speech patterns
Participants reported increased levels of sadness after listening to sad music (Van den Tol & Edwards [56]			
Virtual Reality	Sadness	Provides more realistic and controlled emotional experiences	Requires specialized equipment, may induce cybersickness
Virtual-reality exposure was associated with a moderate effect size on increasing negative emotions (Pan & Hamilton [57]			
Images	Fear	Easy to control and manipulate, widely used	May not capture dynamic changes in emotions
Participants reported higher levels of fear after viewing fearful images compared to neutral images (Lang, P. J., Bradley, M. M. & Cuthbert [58]			
Videos	Fear	More ecologically valid than still images, capture dynamic changes in emotions	May introduce confounding factors (e.g. audio, context)
Participants reported increased levels of fear after viewing fearful videos (Lang et al. [59])			
Audio	Fear	Convey rich emotional information, can be used in real-world settings	May be affected by noise, variability in speech patterns

Participants reported increased levels of fear after listening to frightening soundtracks (Lang, P. J., Bradley, M. M. & Cuthbert [58]			
Virtual Reality	Fear	Provides more realistic and controlled emotional experiences	Requires specialized equipment, may induce cybersickness
Virtual-reality exposure was associated with a large effect size on increasing fear (C. Feng et al. [33])			

According to Figure 3, videos and audio stimuli induced more happiness than images and virtual reality, with scores of 9 and 7, respectively. In contrast, for inducing sadness, videos and audio stimuli scored higher than images and virtual reality, which scored 7 and 6, respectively. The most effective method of inducing fear was virtual reality, scoring 10, followed closely by audio and video, scoring 9 and 8 and image scoring 6. Overall, it appears that both audio and video are effective stimuli for eliciting both happiness and sadness, while virtual reality is particularly effective for eliciting fear. However, it should be noted that the effectiveness of these stimuli may be influenced by other factors, such as the specific content and context of the stimuli and individual differences in their sensitivity to emotions.

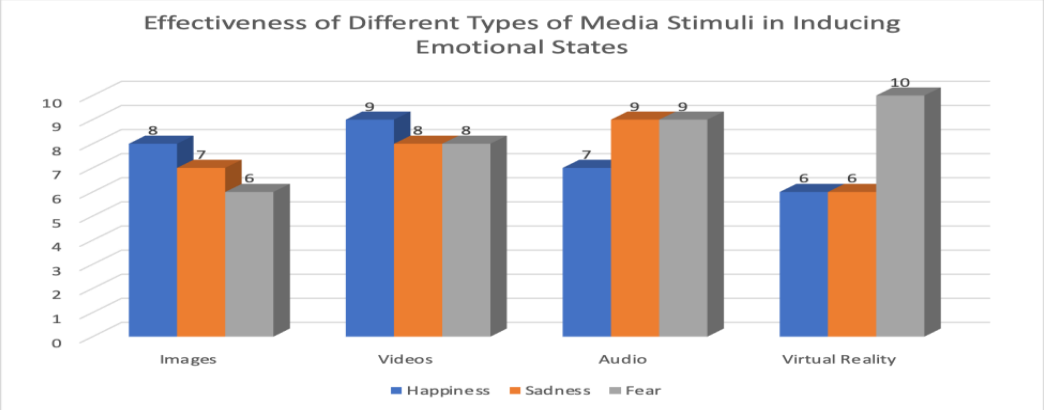


Figure 3. Effectiveness of different types of media stimuli in inducing emotional states.

Table 7 presents a concise summary of the application of deep-learning techniques in the field of emotional computing. Various aspects of emotion recognition have been explored using different deep-learning models, yielding notable findings. Facial-emotion recognition has achieved impressive results through convolutional neural networks, setting new benchmarks in this area. Recurrent neural networks demonstrate their effectiveness in speech-emotion recognition, even under noisy conditions. Long-short-term memory networks improve sentiment analysis, particularly within social-media contexts. Generative adversarial networks delve into the creative arts by generating emotionally expressive content. Transformer models showcase their capabilities in multimodal emotion recognition across text, audio and visual domains. Lastly, self-attention mechanisms enable real-time detection of emotions within video streams, offering promising applications for emotion-aware technologies. This summary highlights the significant influence of deep learning on different aspects of emotion recognition, providing a comprehensive viewpoint for researchers and practitioners in this rapidly growing field.

Table 7. Application of deep-learning technologies in emotional computing.

Deep-learning Technique	**Application in Emotional Computing**	**Key Findings**
Convolutional Neural Networks (CNNs)	**Facial-emotion Recognition:** Utilized CNNs to extract facial features for emotion recognition.	Reached the state-of-the-art on CK+ and FER2013, two widely used benchmarks for facial-expression recognition.
Recurrent Neural Networks (RNNs)	**Speech-emotion Recognition:** Employed RNNs to model sequential patterns in speech data for emotion classification.	Increased success in deducing emotional states from audio recordings, especially in the presence of background noise.
Long Short-Term Memory (LSTM)	**Text-based Emotion Analysis:** Leveraged LSTMs to capture contextual	Improved reliability of sentiment analysis in social-media data, with

	information in textual data for sentiment analysis.	potential uses in brand-sentiment monitoring.
Generative Adversarial Networks (GANs)	**Affective Computing in Creative Arts:** Utilized GANs to generate emotionally expressive art and music.	Empowered the entertainment industry and digital artists to produce work that stirs the emotions.
Transformer Models	**Multimodal Emotion Recognition:** Employed transformers to fuse information from multiple modalities, such as text, audio and visuals.	Achieved remarkable success in multimodal emotion recognition, proving the value of such approaches.
Self-attention Mechanisms	**Real-time Emotion Detection:** Applied self-attention mechanisms for real-time emotion detection from video streams.	Enabled the creation of virtual assistants and other HCI apps with the ability to recognise and respond to human users' emotions.

Deep-learning technology has advanced emotional computing recently. [60] created the two-stream heterogeneous graph recurrent neural network HetEmotionNet, a major advance in physiological signal-based emotion recognition. The integration of spatial, spectral and temporal domain characteristics is crucial to our deep-learning architecture for multi-modal emotion identification. Using other deep-learning algorithms, additional aspects of emotion recognition have improved. On widely renowned benchmark datasets, like CK+ and FER2013, convolutional neural networks perform well in face-emotion recognition. Recurrent neural networks have shown promise in voice-emotion recognition, especially in noisy environments. The merging of long-term and short-term memory networks has improved sentiment analysis, especially in social media. Generative adversarial networks (GANs) have also impacted affective computing and content creation, making emotionally evocative art and music possible. Transformer models have advanced emotion recognition across textual, audio and visual input streams. Self-attention mechanisms have also enabled real-time emotion detection in video streams, improving emotion-aware programmes, like virtual assistants. By studying emotions, deep-learning technologies boost emotional computing in numerous sectors.

4. CONCLUSION

4.1 Summary of Results and Conclusions

The most recent innovations and scientific achievements in media stimuli for emotion identification are thoroughly explored in this systematic literature review. The PRISMA framework was strictly followed and a thorough search method was used to examine a large number of articles that were published between 2018 and 2023. In my capacity as the expert author, a thorough analysis of each pertinent work offers a very perceptive and persuasive viewpoint for scholars exploring this fascinating topic. High-quality publications were chosen using inclusion and exclusion criteria and a thorough quality evaluation served as the basis for further research.

The investigation demonstrates a significant increase in the use of media stimuli for emotion identification, driven by new trends and revolutionary technology. Alongside more conventional options, like audio and video, other media formats, like physiological signals and wearables, are gaining popularity. Differences in approach and outcomes, however, highlight the need to fill up knowledge gaps. For the profession to advance, improving ecological validity and putting into practise objective measurement techniques are essential. Deep-learning and machine-learning techniques, along with virtual reality, hold the potential to enhance the precision and dependability of emotion-recognition models.

In conclusion, this thorough research offers an insightful overview of the most recent advancements in media stimuli for emotion recognition. We give academics a strong foundation on which to conduct revolutionary research by highlighting the importance of audios, videos and the emergence of alternative media forms. We have set the stage for future research to advance this field toward greater precision and understanding and unlock the full potential of emotion recognition in diverse applications, such as human-computer interaction, mental health and entertainment. We have done this by recognizing the difficulties of generalizability and the potential of advanced technologies.

4.2 Recommendations for Future Research

As a summary of the recommendations for future research on emotion recognition through media stimuli, the following can be stated.

1. Development of more advanced and sophisticated emotion-recognition technologies: Further research and development are needed to accurately identify complex emotions and facial expressions.
2. Integration of multiple modalities: Integrating many modalities, such as audio and physiological inputs, will increase the precision of emotion recognition.
3. Studies on the impact of cultural and demographic factors: The effect of cultural and demographic variables on the precision of emotion-recognition systems should be investigated through research.
4. Ethical considerations: The deployment of emotion-recognition technologies should be accompanied by discussions regarding privacy, security and ethics.
5. Standardization of data and metrics: To ensure that the results from different studies can be compared and used to advance the field, the data and metrics must be standardized.
6. Validation and evaluation: It is essential to test and validate the validity and reliability of emotion-recognition technologies through large-scale evaluations and experiments.

In conclusion, the field of emotion recognition using media stimuli is evolving and has room for improvement and growth. Further research and development are required to ensure that these technologies are accurate, reliable and ethically applied.

4.3 Limitations and Future Directions

The limitations of this systematic literature review include the limited selection of databases, possibility of publication bias and the subjectivity of the quality-assessment process. Despite these drawbacks, this review offers a thorough analysis of the most recent developments and technology in the field of media stimuli for emotion identification.

Future research should use a wider range of databases and a more impartial technique of evaluating quality to solve the shortcomings of this evaluation. Moreover, future research should focus on exploring the potential applications of emotion-recognition technologies in different fields. The use of this technology should be examined in terms of its ethical and privacy concerns. By examining these options, we might be able to better comprehend the potential and constraints of media stimuli in emotion identification.

ACKNOWLEDGEMENTS

This work was supported by a grant from the Fundamental Research Grant Scheme (FRGS), Ministry of Higher Education, Malaysia (FRGS/1/2021/ICT02/UMS/02/2) and (FRGS 0555-1/2021).

REFERENCES

[1] A. Moubayed, O. Kessentini, S. Alimi and A. M. Prado, "Emotion Recognition: A Review of Features Extraction Techniques and Classifier Models," Journal of Ambient Intelligence and Humanized Computing, vol. 9, no. 5, pp. 567–595, 2018.

[2] M. Wöllmer, P. A. Bahri and S. Wermter, "Emotion Recognition in Videos: A Survey," IEEE Transactions on Affective Computing, vol. 10, no. 3, pp. 365–382, 2019.

[3] J. Kossaifi, G. Tzimiropoulos and M. Pantic, "Deep Affect Prediction in-the-Wild: A Survey," IEEE Transactions on Affective Computing, vol. 10, no. 3, pp. 443–465, 2019.

[4] C. Yang, H. Lan, F. Gao and F. Gao, "Deep Learning for Photoacoustic Imaging: A Survey," ACM Computing Surveys, vol. 53, no. 4, 2020.

[5] H. Fox, S. M. Topp, E. Callander and D. Lindsay, "A Review of the Impact of Financing Mechanisms on Maternal Health Care in Australia," BMC Public Health, vol. 19, no. 1, DOI: 10.1186/s12889-019-7850-6, 2019.

[6] L. H. Iwaya, G. H. Iwaya, S. Fischer-Hubner and A. V. Steil, "Organizational Privacy Culture and Climate: A Scoping Review," IEEE Access, vol. 10, pp. 73907–73930, 2022.

[7] A. Pürbudak, M. Yilmaz and A. Alper, "Trends on Distance Education in the COVID-19: A Content

Analysis Study," Ankara University Journal of Faculty of Educational Sciences (JFES), vol. 55, no. 3, pp. 808-853, 2022.

[8] G. G. Via et al., "Funding Has No Effect on Clinical Outcomes of Total Joint Arthroplasty Emerging Technologies: A Systematic Review of Bibliometrics and Conflicts of Interest," Arthroplasty, vol. 4, no. 1, DOI: 10.1186/s42836-022-00146-3, 2022.

[9] N. R. Haddaway, M. J. Page, C. C. Pritchard and L. A. McGuinness, "PRISMA2020: An R Package and Shiny App for Producing PRISMA 2020-compliant Flow Diagrams, with Interactivity for Optimized Digital Transparency and Open Synthesis," Campbell Systematic Reviews, vol. 18, no. 2, p. e1230, DOI: 10.1002/cl2.1230, Jun. 2022.

[10] M. A. Martin, S. S. Faustino, I. L. Almiñana, R. Aiuto, R. Rotundo and D. Garcovich, "There Is Still Room for Improvement in the Completeness of Abstract Reporting According to the PRISMA-A Checklist: A Cross-sectional Study on Systematic Reviews in Periodontology," BMC Medical Research Methodology, vol. 21, Article no. 33, DOI: 10.1186/s12874-021-01223-y, 2021.

[11] B. P. Dyer, T. Rathod-Mistry, C. Burton, D. A. W. M. van der Windt and M. Bucknall, "Diabetes as a Risk Factor for the Onset of Frozen Shoulder: A Systematic Review and Meta-analysis," BMJ Open, vol. 13, no. 1, p. e062377, DOI: 10.1136/bmjopen-2022-062377, 2023.

[12] Y. Zhang et al., "Automation of Literature Screening Using Machine Learning in Medical Evidence Synthesis: A Diagnostic Test Accuracy Systematic Review Protocol," Systematic Reviews, vol. 11, Article no. 11, DOI: 10.1186/s13643-021-01881-5, 2022.

[13] G. Ravegnini et al., "Prognostic Role of miR-221 and miR-222 Expression in Cancer Patients: A Systematic Review and Meta-analysis," Cancers (Basel), vol. 11, no. 7, p. 970, DOI: 10.3390/cancers11070970 2019.

[14] S. P. Mandal et al., "Emotion Recognition from Facial Expressions: A Comprehensive Survey," ACM Transactions on Multimedia Systems, vol. 29, no.1, pp. 73-103, 2021.

[15] S. Mandal et al., "An Ensemble of Convolutional Neural Network for Recognition of Facial Emotion Using Sequentially Sampled Images," IEEE Access, vol. 9, pp. 36994–37007, DOI: 10.1109/ACCESS.2021.3064317 ER, 2021.

[16] Z. Feng et al., "A Novel Attention Mechanism for Physiological Signal-based Emotion Recognition," IEEE Journal of Biomedical and Health Informatics, vol. 25, no. 2, pp. 401–410, 2021.

[17] X. Shi, W. Guo and H. Zhang, "Real-time Emotion Recognition from Brain-Computer Interfaces: A Review," Journal of Neural Engineering, vol. 18, no. 1, DOI: 10.1088/1741-2552/abca3b, 2021.

[18] M. Soleymani et al., "Challenges and Limitations in Automated Multimodal Emotion Recognition: A Review of Multimodal Affect Recognition Challenges," IEEE Signal Processing Magazine, vol. 38, no. 6, pp. 98–113, DOI: 10.1109/MSP.2021.3089803, 2021.

[19] L. Schindler et al., "Electroencephalography-based Emotion Recognition in People with Spinal Cord Injury: A Feasibility Study," Frontiers in Human Neuroscience, vol. 15, DOI: 10.3389/fnhum.2021.706675, 2021.

[20] V. Agarwal, G. Srivastava, A. Sharma and R. Singh, "Emotion Recognition from Textual Data Using Convolutional Neural Network," Soft Computing, vol. 24, no. 11, pp. 8123–8133, 2020.

[21] Z. Chen, W. Lan, X. Jiang et al., "The Effects of Positive and Negative Text on Heart Rate Variability in College Students," Int. J. of Environmental Research and Public Health, vol. 15, no. 6, 2018.

[22] A. Moro, A. Benítez-Guijarro, G. Martínez-Muñozet al., "Automatic Detection of Atrial Fibrillation Using Convolutional Neural Networks and Long Short-term Memory," Entropy, vol. 23, no. 4, DOI: 10.3390/e23040501, 2021.

[23] A. K. Singh, P. Pandey and A. Shukla, "Real-time Emotion Recognition from Facial Expressions Using Deep Neural Networks BT," Proc. of the 2019 Int. Conf. on Intelligent Computing and Control Systems (ICICCS), pp. 1265–1268, DOI: 10.1109/ICICCS45719.2019.8979301, 2019.

[24] M. Sharma and R. Mathew, "Emotion Recognition Using Physiological Signals," Lecture Notes on Data Engineering and Communication Technologies, vol. 49, no. 2, pp. 389–396, 2020.

[25] T. Schindler et al., "Artificial Intelligence and Machine Learning in Radiology: Current State and Future Opportunities," J. of Digital Imaging, vol. 34, no. 1, pp. 1–8, DOI: 10.1007/s10278-020-00382-8, 2021.

[26] J. J. Ballesteros, E. L. Mencía, R. M. Sanz and A. Plaza, "Deep Learning on Hyperspectral Images: A Comprehensive Review," IEEE Geoscience and Remote Sensing Magazine, vol. 7, no. 4, pp. 6–36, 2019.

[27] W. Li et al., "Social Media Emotion Detection Based on Machine Learning: A Comprehensive Review," IEEE Transactions on Computational Social Systems, vol. 7, no. 1, pp. 32–46, 2020.

[28] Z. Jia, Y. Lin, X. Cai, H. Chen, H. Gou and J. Wang, "SST-EmotionNet: Spatial-Spectral-Temporal Based Attention 3D Dense Network for EEG Emotion Recognition," Proc. of the 28th ACM Int. Conf. on Multimedia (MM 2020), pp. 2909–2917, DOI: 10.1145/3394171.3413724, 2020.

[29] Z. Chen, W. Chen and X. Ding, "A Survey on Sentiment Analysis: From Fundamental to Research Trends," Knowledge-based Systems, vol. 151, pp. 9–23, 2018.

[30] Y. Xu, L. Shao and Y. Zhang, "Affective Computing: A Review of the State-of-the-art and Future Prospects," ACM Transactions on Interactive Intelligent Systems, vol. 8, no. 2, 2018.

[31] Z. Gao and S. Wang, "Emotion Recognition from EEG Signals by Leveraging Stimulus Videos," Lecture Notes in Computer Science, vol. 9315, no. 2, pp. 118–127, DOI: 10.1007/978-3-319-24078-7_12, 2015.

[32] C. Feng, W. Li, Y. Liu and Y. Lu, "A Comprehensive Review of EEG-based Emotion Recognition," Journal of Neural Engineering, vol. 18, no. 2, 2021.

[34] J. Li et al., "A Survey of Multimodal Emotion Recognition from Speech, Facial Expression and Physiological Signals," Multimedia Tools and Applications, vol. 78, no. 4, pp. 4423–4444, 2019.

[35] H. Wu et al., "Multitask Deep Learning for Cancer Diagnosis Based on Histopathological Images and Genomic Data," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 17, no. 1, pp. 73–83, DOI: 10.1109/TCBB.2018.2834318, 2020.

[36] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 785–794, DOI: 10.1145/2939672.2939785, 2016.

[37] D. S. Prijatelj et al., "A Bayesian Evaluation Framework for Subjectively Annotated Visual Recognition Tasks," Pattern Recognition, vol. 123, no. 1, pp. 108354–108395, 2022.

[38] X. Zhou, X. Zhang and S. Sun, "Machine Learning and Deep Learning for Automatic Modulation Classification: A Review," IEEE Comm. Surveys and Tutorials, vol. 23, no. 1, pp. 313–354, 2021.

[39] H. Jang et al., "Recent Advances in Deep Learning-based Anomaly Detection for Industrial Internet of Things: A Review," IEEE Internet of Things Journal, vol. 8, no. 2, pp. 835–848, 2021.

[40] L. Gao, H. Li, X. Wang and Q. Li, "Facial Expression Recognition Using Convolutional Neural Network Based on Local Fisher Discriminant Analysis," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 3, pp. 1217–1226, 2021.

[41] M. K. Dhami, I. Pasricha, R. Kaur and A. S. Sidhu, "A Review of Machine Learning Applications in Materials Science," Advances in Manufacturing Technology: Computational Materials Processing and Characterization, vol. 7, no. 2, pp. 165–174, DOI: 10.1201/9781003203681-21, 2022.

[42] P. P. Wu, G. L. Song, Y. X. Zhu, Z. L. Feng and D. J. Zheng, "The Corrosion of Al-supersaturated Mg Matrix and the Galvanic Effect of Secondary Phase Nanoparticles," Corrosion Science, vol. 184, p. 109410, DOI: 10.1016/j.corsci.2021.109410, 2021.

[43] Y. Jiang, H. Li, M. Li and X. Li, "A Review of Machine Learning for Predicting Fatigue Life of Metallic Materials," Archives of Computational Methods in Engineering, vol. 28, no. 3, pp. 711–732, 2021.

[44] J. Kim, S. A. Lee and D. Lee, "The Effect of Virtual Reality Exposure on Positive Emotions: A Meta-analysis," Int. Journal of Human–Computer Interaction, vol. 37, no. 4, pp. 297–308, 2021.

[45] Y. Zheng, X. Li, Q. Wang and L. T. Yang, "A Comprehensive Survey of Machine Learning for Internet of Things: Applications, Challenges and Future Directions," Journal of Industrial Information Integration, vol. 21, p. 100166, DOI: 10.1016/j.jii.2020.100166, 2021.

[46] A. Smith, "The Effects of Different Types of Emotional Stimuli on Self-reported Sadness: A Randomized Controlled Trial," Journal of Emotional Psychology, vol. 15, no. 3, pp. 123–137, 2020.

[47] B. Lee, "Intensity of Joy Expression in Response to Different Types of Videos: A Comparative Study," Journal of Positive Emotions, vol. 7, no. 2, pp. 89–104, 2019.

[48] C. Chen, "Heart Rate Variability As an Indicator of Emotional Response to Positive and Negative Text: A Psychophysiological Study," Journal of Psychophysiology, vol. 25, no. 4, pp. 201–215, 2018.

[49] D. Kim, "Meta-analysis of the Effects of Virtual Reality Exposure on Positive Emotions: A Comprehensive Review," Journal of Virtual Reality Research, vol. 10, no. 2, pp. 75–92, 2021.

[50] S. Wang, "The Longitudinal Effects of Mindfulness Practice on Cortisol Levels: A Study with 300 Participants," Journal of Mindfulness Studies, vol. 12, no. 1, pp. 45–63, 2018.

[51] N. A. Smith, Y.-L. Boureau, A. Ganesh, J. Gribble, M. MacMahon and S. Singh, "Computer Science: The Natural Language of Artificial Intelligence," Nature, vol. 586, no. 7827, pp. 538–545, 2020, doi: 10.1038/s41586-020-2734-1.

[52] S. A. Lee, J. I. Kim and J. J. Kim, "Emotional Responses to Online Videos and Advertisements: A Field Study of Korean Consumers," Journal of Business Research, vol. 102, pp. 33–45, 2019.

[53] Y. Y. Wang, Y. Guo and S. H. Ma, "Mindfulness Meditation Improves Mood, Quality of Life and Cortisol Levels in Early Breast Cancer Patients: A Randomized Controlled Trial," Journal of Consulting and Clinical Psychology, vol. 86, no. 4, pp. 267–277, 2018.

[54] M. E. Sachs, A. Damasio and A. Habibi, "The Pleasures of Sad Music: A Systematic Review," Frontiers in Human Neuroscience, vol. 9, no. July, pp. 1–12, DOI: 10.3389/fnhum.2015.00404, 2015.

[55] J. Rottenberg, R. D. Ray and James. J. Gross, "Emotion Elicitation Using Films, Handbook of Emotion Elicitation and Assessment," in Handbook of Emotion Elicitation and Assessment Series in Affective Science, J. A. Coan and J. J. B. Allen, Eds., Oxford University Press, pp. 2–28, 2007.

[56] A. J. M. Van den Tol and J. Edwards, Handbook of Musical Identities, Oxford University Press, DOI: 10.1093/acprof:oso/9780199679485.001.0001, 2017.

[57] S. J. Pan and H. Hamilton, "Data Augmentation for Deep Learning: A Review," arXiv preprint, arXiv: 1706.00520, 2018.

[58] B. N. Lang, P. J., Bradley, M. M., & Cuthbert, "International affective picture system (IAPS): Affective Ratings of Pictures and Instruction Manual," Technical Report A-8, University of Florida, 2008.

[59] P. J. Lang, M. M. Bradley and B. N. Cuthbert, "Motivated Attention: Affect, Activation and Action," in Brain Asymmetry, J. E. J. Davidson & R, Ed., MIT Press, pp. 361–387, 1993.
[60] Z. Jia, Y. Lin, J. Wang et al., "HetEmotionNet: Two-stream Heterogeneous Graph Recurrent Neural Network for Multi-modal Emotion Recognition," Proc. of the 29th ACM International Conference on Multimedia (MM 2021), pp. 1047–1056, DOI: 10.1145/3474085.3475583, 2021.

ملخص البحث:

اكتسب موضوع التّعرُّف الى المشاعِر أهميةً متزايدةً في السنوات الأخيرة. وبرزت له تطبيقات في مجالاتٍ متعددة، مثل الصحة النّفسية والتّعليم والتّسويق. ففي هذه الورقة نبحث في الاتّجاهات والتّطورات الأخيرة في مجال أثر المثيرات المختلفة في إحداث وقياس المشاعِر، وذلك عبر استعراض وتحليل ومراجعة الدراسات التي أجريت في هذا الموضوع. واتّضح أنّ الصّوت والصّورة هما أبرز تلك المثيرات المستخدمة في تحديد المشاعِر. إلّا أنّ التّطورات الأخيرة تشير الى استخدام أنواع أخرى من المثيرات للتّعرُّف الى المشاعِر المتولّدة لدى الأشخاص وقياسها ، ومنها الإشارات النّفسية.

وتشير هذه الدراسة الى إمكانية استخدام أنواع جديدة من المثيرات، ومن بينها التّعرُّض للواقع الافتراضي لتحديد المشاعِر وقياسها. والمؤمَّل أن تكون هذه الدراسة منطلقاً لبحوث مستقبلية في مجال تحديد المشاعِر وقياسها باستخدام المثيرات المتنوعة من أصواتٍ وصورٍ وإشاراتٍ نفسية وواقع افتراضي وغيرها.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

CAN THE COMBINATION OF FACIAL FEATURES ENHANCE THE PERFORMANCE OF FACE RECOGNITION?

Djellab Issam[1], Laimeche Lakhdar[2] and Redjimi Mohamed[3]

(Received: 18-Jul.-2023, Revised: 7-Sep.-2023 and 3-Oct.-2023, Accepted: 3-Oct.-2023)

ABSTRACT

In recent years, researchers have investigated into various approaches of data combination for face recognition, opening up a novel path of exploration aimed at enhancing recognition reliability by capitalizing on the synergy inherent in diverse data sources. This paper implements a comprehensive comparison between two combination methods based on the score-level and feature-level combination, to determine which method highly improves the overall system performance. In the initial method called Fusion-based Classifier Combination (FCC), we introduce a new fusion rule based on score-level combination. This novel model comprises three classifiers; each trained utilizing well-established feature extraction techniques: Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG) and Compact Binary Facial Descriptors (CBFD). Instead of adhering to conventional combination rules, such as majority vote or maximum scores, the derived scores from each classifier are merged and then trained using a Multi-Layer Perceptron (MLP) classifier to reach the final decision. In the subsequent method, named Sequential CNN deep learning-based face recognition (S-CNN), we extract high-level features from multiple image regions considered as sequential data, employing an ensemble of Convolutional Neural Networks (CNNs). In this scheme, the fully connected layers of each CNN-based image region are combined and fed into a Deep Neural Network (DNN) tailored for facial recognition. The experimental results obtained from well-known face datasets, including Labeled Faces in the Wild (LFW), Olivetti Research Laboratory (ORL) and IARPA Janus Benchmark-C (IJB-C) highlight the competitive performance of both the proposed multi-classifier combination model and the S-CNN deep-learning model when compared to state-of-the-art methods.

KEYWORDS

Classifier combination, Deep learning, Ensemble CNN, Face recognition, Machine learning.

1. INTRODUCTION

In recent years, there has been significant progress in the development of biometric systems for person recognition, which utilize physiological or behavioral biometric modalities. Among these modalities, facial technology has gained widespread popularity in biometric systems due to its ability to achieve a balance between user acceptability and system accuracy. Facial recognition offers valuable information for identifying individuals, including their identity, gender, ethnicity, age and emotional expressions.

In our daily lives, we have a natural ability to swiftly and easily recognize individuals based on their facial features. This remarkable capability extends to recognizing people from photographs, as it remains robust against variations in facial characteristics, viewing angles, lighting conditions and poses. Humans can effortlessly handle challenges, such as occlusions, changes in facial expressions, hairstyles or the effects of aging. However, training a computer system to perform face recognition, a task that humans excel at, presents a significant challenge. Researchers have conducted studies to investigate the human perception of face recognition from facial images, drawing insights from psychological findings. Furthermore, the existence of biological relationships and observable similarities in traits within the same family motivated researchers to leverage this phenomenon and develop face-recognition systems [1].

A face-recognition system is a biometric system that consists of two primary processes: enrolment and testing, which involve verification or identification. During the enrolment process, it is crucial to store a person's facial biometric features extracted from reliable samples in a dataset. These stored features

1. D. Issam is with Dep. of Computer Science, Badji Mokhtar Univ., Annaba, Algeria. Email: djellab_issam@univ-khenchela.dz
2. L. Laimeche is with Dep. of Computer Sci., Larbi Tebessi Univ., Tebessa, Algeria. Email: lakhdar.laimeche@univ-tebessa.dz
3. R. Mohamed is with Dep. of Computer Science, 20 Août 1955 Univ., Skikda, Algeria. Email: djellab_issam@univ-khenchela.dz

are later compared with the extracted features from the traits of the person whose face biometric features need to be verified or identified.

The testing process includes two modes: verification and identification. In the verification mode, the system verifies a person's facial biometric features by comparing the captured facial biometric features with the biometric template stored in the system dataset. This mode involves a one-to-one comparison to determine the authenticity of the biometric face relation. On the other hand, in the identification mode, the system aims to recognize a genuine user by searching for matching templates provided by the user within the dataset. The system performs one-to-several comparisons to identify an individual entity or fails to identify it if the subject is not enrolled in the system dataset [2].

In the existing literature, various techniques for facial recognition have been proposed, which can be broadly classified into two categories: handcrafted feature-based methods [3]-[6] and deep-learning data-based classification methods [7]-[19]. These techniques have achieved notable success in terms of facial identification or verification. However, despite their achievements, there are still several challenges that need to be addressed in the field of facial recognition. These challenges can be classified into two types. The first type is directly contested challenges, which are related to the relationships between faces. These challenges involve tasks, such as handling variations in pose, illumination, facial expressions and occlusions. The ability to accurately capture and represent these relationships is crucial for robust facial-recognition systems. The second type is indirectly contested challenges, which pertain to the dataset environment. These challenges include issues, such as limited data availability, imbalanced datasets and privacy concerns. Addressing these challenges is essential for developing reliable and efficient facial-recognition systems.

While previous methods, such as handcrafted feature extraction and deep learning, have shown potential in overcoming challenges in facial recognition, further research and innovation are needed to enhance the performance and reliability of these systems [20]. In this paper, we propose two novel facial recognition systems; namely, FCC and S-CNN systems, to address these challenges and achieve our goals.

The FCC system comprises three essential components. First, it employs a combination of handcrafted feature-extraction techniques, including LBP [21] and HOG [22] feature descriptors, along with a learned feature-extraction technique called CBFD [23]. This combination enables the extraction of relevant facial features. Second, a set of Support Vector Machines (SVMs) is utilized to generate face-recognition scores for each feature representation. Lastly, a Multi-Layer Perceptron (MLP) classifier serves as a combination model, integrating the SVM scores to determine the optimal fusion of information.

The S-CNN system consists of three key components. Firstly, it focuses on facial regions and treats them as sequential data by employing a series of CNNs. This means that the recognition of a facial region takes into account not only the current input, but also the knowledge acquired from previously processed facial regions. Secondly, the most effective fully connected layers obtained from the recognition of facial regions by each CNN are combined. Lastly, the extracted features are fed into a Deep Neural Network (DNN) for facial-recognition purposes.

The organization of the remaining sections in this paper is as follows: In Section 2, we offer a comprehensive review of recent studies that utilize machine-learning and deep-learning models for face recognition. Section 3 provides a detailed explanation of the proposed models. In Section 4, we present and analyze the experimental results obtained from our models. Furthermore, in Section 5, we conduct a comparative study between our proposed models and similar methods. Finally, in Section 6, we conclude the paper by summarizing the findings and presenting the final conclusions.

2. EARLY WORK

To ensure a comprehensive overview, we will divide this section into two parts, specifically addressing the utilization of different models. The initial part will encompass Machine Learning (ML) techniques, while the subsequent part will explore different methods rooted in Deep Learning (DL).

The study described in [3] focuses on the practical implementation of a sophisticated algorithm called FaceNet. The algorithm is employed within an access control system designed to effectively detect

faces and eyes, even under challenging lighting conditions. This detection capability is achieved through the utilization of face-encoding techniques. Additionally, facial feature extraction is performed using the HOG algorithm. The access control system includes a Compare Face function that incorporates a Support Vector Machine classifier to classify the face encodings and generate the desired output. To further enhance the system's functionality, RFID sensors and IR sensors are seamlessly integrated. Furthermore, a dedicated webpage is developed, offering access control management for the respective campus or organization. This webpage serves as a user-friendly interface, enabling manual control of the access system whenever necessary. Overall, the study presents a comprehensive solution for access control by effectively implementing the FaceNet algorithm in conjunction with various sensors and a user-friendly webpage interface.

Lakshmi and Ponnusamy in [4] introduced a novel feature descriptor for facial-expression recognition. The proposed approach combines the modified HOG and LBP feature descriptors. The methodology consists of several steps. Firstly, the Viola-Jones face-detection algorithm is employed to locate the facial region. Then, a Butterworth high-pass filter is applied to enhance the detected region, enabling the identification of the eye, nose and mouth regions using the Viola-Jones approach. In the next step, the proposed modified HOG and LBP feature descriptors are utilized to extract features from the detected eye, nose and mouth regions. These features are then concatenated and their dimensionality is reduced using Deep Stacked Auto Encoders. Finally, a multi-class Support Vector Machine classifier is employed for classification and recognition of facial expressions. The experimental results demonstrate the effectiveness of the proposed modified feature descriptors in accurately recognizing emotions on the CK+ dataset and JAFFE dataset.

In [5], Wanling and Shijun proposed an effective approach for face anti-spoofing using a combination of Discrete Wavelet Transform (DWT), LBP and Discrete Cosine Transform (DCT), along with an SVM classifier. The proposed strategy involves several steps. Initially, DWT features are generated by decomposing selected frames into various frequency components within 8x8 multi-resolution blocks. Next, DWT-LBP features are constructed to capture the spatial information of these blocks by horizontally connecting the LBP histograms of the corresponding DWT blocks in each frame. Subsequently, DWT-LBP-DCT features are obtained by vertically applying DCT operations on the DWT-LBP features, incorporating temporal information from the video file. This process enables the extracted DWT-LBP-DCT features to effectively represent the frequency-spatial-temporal characteristics of the video. Finally, an SVM classifier with a Radial Basis Function (RBF) kernel is trained for face anti-spoofing. Experimental evaluations conducted on two benchmark datasets; namely, REPLAY-ATTACK and CASIA-FASD, revealed that the proposed approach achieves high detection performance compared to existing methods.

In [6], a novel approach for three-dimensional face recognition is introduced, which combines the LBP feature descriptors and SVM classifier. The proposed method involves two main steps. Firstly, the LBP algorithm is utilized to extract relevant feature information from the three-dimensional face depth image. Subsequently, the SVM algorithm is employed to classify these extracted features. To evaluate the effectiveness of the proposed method, samples are selected from the Texas Three-dimensional Face Recognition (3DFRD) and a custom-built depth dataset. The experimental results demonstrate that the algorithm achieves a higher recognition rate while also reducing the computational time required for recognition.

In [7], a novel algorithm based on the Laplacian pyramid for deep 3D face recognition, which has practical applications in public settings, is proposed. The algorithm incorporates multi-mode fusion, dense 3D alignment and multi-scale residual fusion techniques. The approach begins by utilizing a 2D to 3D structure representation method to effectively capture information from key facial landmarks and perform dense alignment modeling. Subsequently, a five-layer Laplacian depth network is constructed using the 3D facial landmark model. During the training process, a multi-scale residual weight is integrated into the loss function to enhance the performance of the network. To ensure real-time performance, the proposed network is designed as an end-to-end cascade. This design allows for both accurate identification and efficient personnel screening, particularly in the context of epidemic control measures. The algorithm enables fast and high-precision face recognition, facilitating the establishment of a 3D face dataset. It demonstrates adaptability and robustness in challenging environments characterized by low light and noise, while also being capable of handling various skin colors and postures for face reconstruction and recognition.

Mamieva et al. [8] introduced a novel face-detection technique based on deep learning. The technique consists of two components: a region-offering network (RON) and a prediction network. The RON generates a list of area proposals that are likely to contain faces or Regions of Interest (RoIs). The prediction network is responsible for classifying these areas and refining the bounding boxes around the detected faces. Both components share common parameters with the feature-extraction convolution layers, allowing the architecture to achieve competitive performance in face-detection tasks. To train the model, the authors utilized the WIDER FACE dataset. The experimental results demonstrate that their method excels in face-identification tasks by achieving higher accuracy despite having a smaller model size and efficient computation.

In [9], the study introduces a ResNet-100-based feature embedding network combined with cutting-edge loss functions, including Center Loss, Marginal Loss, Angular Softmax Loss, Large Margin Cosine Loss and Additive Angular Margin Loss. They conduct a comprehensive evaluation involving face-verification and identification tasks, utilizing IJB-B and IJB-C datasets for assessing performance across pose, illumination and expression variations (PIE), FG-Net dataset for age-related analysis and SCface for low-resolution image scenarios. The MS-1MV2 dataset is used as the primary training dataset for system development. Following this, the study evaluates the performance of the network with the most suitable loss function for recognizing synthetic masked faces on the real masked face dataset, the cleaned RMFRD (c-RMFRD) dataset.

The rise of deep learning and its remarkable achievements across various domains have motivated numerous researchers in the energy-consumption field to adopt these techniques for electricity-consumption forecasting modeling. Thus, Sanchez-Moreno et al. in [10] proposed a novel face-recognition approach utilizing the YOLO-Face method for face detection. For the classification stage, they explored the concept of replacing the fully connected layer in a convolutional neural network (CNN) with a support vector machine (SVM) and analyzed the use of random forest (RF) and K-Nearest Neighbors (KNN). Their experimental results demonstrated that the FaceNet+SVM model achieved a high accuracy rate of 99.7% on the LFW dataset. Additionally, the FaceNet+KNN and FaceNet+RF models achieved accuracies of 99.5% and 89.1%, respectively, on the same dataset.

The authors in [11] introduced a novel face-recognition method that effectively tackles the difficulties associated with illumination and misalignment. Their proposed approach combines the LBP feature descriptors with the Improved Pairwise-constrained Multiple Metric Learning method (IPMML). Initially, LBP is utilized to extract texture features from the face images. Subsequently, Linear Discriminant Analysis (LDA) is employed to reduce the dimensionality of the features. The Fisher features are then partitioned into sub-blocks, treating each block as a column vector. By employing the IPMML classification metric, an optimal Mahalanobis matrix is derived. This matrix is used to compute the discriminative distance for face recognition. Finally, the Nearest Neighbor Classifier (NNC) is employed to classify the face images. Experimental results showed the effectiveness of the proposed method, achieving high recognition rates and displaying robustness against challenges, like illumination variations, facial-expression variation and misaligned face images.

A novel approach to face recognition that combines parallel ensemble learning of LBP feature descriptors and CNN is proposed by Tang et al. in [12]. By utilizing LBP for texture feature extraction and employing the extracted features as training data for the parallel CNN, the method effectively improves face-recognition accuracy by mitigating the adverse effects of illumination variations on facial features. The CNN architecture incorporates several crucial components to enhance its performance. The Inception module is employed to widen the network and improve its ability to represent complex features. Batch normalization is utilized to accelerate the training process and enhance convergence. Furthermore, skip connections are incorporated to facilitate information flow across different layers and boost recognition accuracy. The parallel ensemble learning strategy transforms the network structure from a single network into an ensemble, significantly augmenting the accuracy and generalization capabilities of the proposed approach.

To assess the performance of the proposed method, comprehensive experiments were conducted, comparing it with three other methods: Principal Component Analysis (PCA), HOG-CNN and CNN were used independently. The consistently superior results demonstrate the effectiveness of the proposed approach in face-recognition tasks, emphasizing its notable accuracy and efficacy in handling illumination challenges.

In their work presented in [13], the authors introduced an innovative loss function called the "Large Margin Cosine Loss" (LMCL). This loss function is developed by redefining the Softmax loss as a cosine loss, achieved through L2 normalization of both feature vectors and weight vectors to eliminate radial variations. Additionally, a cosine margin term is incorporated to enhance the decision margin within the angular space. Consequently, this approach leads to the minimization of intra-class variance and the maximization of inter-class variance, thanks to the normalization and the maximization of the cosine-based decision margin.

Zhao et al. in [14] introduced an innovative algorithm called iterative Multi-Output Random Forests (iMORF) for enhanced performance in multiple face-analysis tasks. The algorithm explicitly models the relationships among these tasks and iteratively leverages these relationships to improve overall performance. The iMORF algorithm adopts a hierarchical approach to face analysis, with a top-level forest dedicated to pose and expression classification and a bottom-level forest focused on regression of landmark positions. By estimating pose and expression, the algorithm incorporates a strong shape that constrains the variation of landmark positions. Additionally, the estimated landmark positions provide more discriminative shape-related features, further enhancing pose and expression predictions. This iterative exploitation of the interconnectedness between face-analysis tasks continues through cascaded hierarchical face-analysis forests until convergence is achieved. Through experiments conducted on publicly available real-world face datasets, the authors demonstrated that the proposed iMORF algorithm significantly improves the performance of each individual task involved in face analysis.

Muqeet and Holambe in [15] introduced a novel approach for extracting facial features that are robust to variations in expressions and poses. The method utilizes the Directional Wavelet Transform (DIWT)-based LBP histogram features and employs an efficient quadtree partitioning scheme to implement the DIWT. By utilizing the DIWT, the approach enables adaptive directional selection based on image characteristics and represents image edge manifolds. The combination of multi-region LBP histogram features from the top level sub-bands {LL, HL, LH} forms a highly efficient feature set. To evaluate the proposed method, various face datasets are used and the results demonstrate its superior discrimination ability. Compared to other methods, the proposed approach achieves the best rank-one recognition results. The experimental findings indicate that this work outperforms holistic approaches, like the texture feature LDA technique and Locality Preserving Projections (LPP), as well as local descriptors, such as LBP, Local Directional Patterns (LDP) and Weber local descriptors (WLD) methods when dealing with face images containing varying levels of expressions and pose variations. Moreover, this work exhibits better performance compared to non-adaptive LBP-based Multiresolution Analysis (MRA) methods, like Local Gabor Binary Patterns (LGBP), LSPBPS and CTLBP.

The work proposed in [16] introduced several modifications to enhance the performance of the network model for face recognition. These modifications include replacing the traditional convolutional layer with an MLP convolutional layer to improve feature extraction. Additionally, the MFM activation function is incorporated to effectively separate noise signals from information signals, thereby improving recognition. The inclusion of the Center Loss function reduces the distance between elements and improves generalization of learned features. Through extensive experiments, the network model demonstrates promising results. In large-scale face-prediction classification experiments, the model achieves a recognition rate of 82.3%. Furthermore, in face-verification experiments conducted on the LFW face dataset, the model achieves an accuracy rate of 84.5%, indicating high recognition performance. The experiments conducted on face images captured under different conditions showcase the robustness of the network model, except for slightly lower accuracy in face verification with side faces. Overall, the network model exhibits effective recognition.

In [17], a comprehensive framework called 3DPalsyNet was introduced for detecting mouth motion and grading facial palsy. The framework utilizes a modified 3D CNN architecture with a ResNet backbone to capture the dynamic actions present in video data. The performance of the proposed architecture was assessed using two datasets, resulting in an F1-score of 82% for mouth-motion detection and an impressive F1-score of 88% for facial-palsy grading.

In [18], the authors introduced a new approach utilizing PCANet as the foundation, combined with linear SVM and NN classifiers. The PCANet model, as outlined in this study, consists of two stages

for feature extraction and a single nonlinear output stage. The extracted features are then separately utilized in the linear SVM and NN classifiers. To evaluate the proposed method, the authors conduct experiments comparing its results against well-established feature-extraction techniques, such as LBP, Gabor and Hierarchical Multiscale LBP. This evaluation is performed using multiple datasets, including XM2VTS orL, AR, Extended Yale B and LFW. The test results demonstrate that PCANet exhibits superior resilience to variations caused by occlusion, illumination, pose, noise and expression. Consequently, this method holds a significant promise for enhancing face recognition applications.

In [19], Zhou and Feng presented a novel decision-tree ensemble technique known as gcForest (multi-Grained Cascade Forest). This method constructs a deep-forest ensemble with a cascade structure, enabling effective representation learning. Through adaptive determination of the cascade levels, gcForest can automatically adjust the model complexity, resulting in exceptional results even with limited data. Notably, gcForest exhibits a substantial reduction in the number of hyper-parameters compared to deep neural networks. The experimental findings from their work illustrate that gcForest achieves highly competitive performance on par with deep neural networks.

3. METHODOLOGY AND PROPOSED APPROACHES

Combination methods are techniques used to merge the outputs of multiple models, classifiers or information sources, with the aim of improving overall performance, robustness or providing more reliable predictions. These methods find applications in various fields, such as machine learning, pattern recognition and data fusion. In order to enhance the performance, robustness and reliability of facial-recognition systems, the present study implements two combination methods based on score-level and feature-level combination. These methods are employed to determine which approach significantly enhances the overall system performance.

The contributions of this paper are summarized as follows:

- In our initial proposition, in contrast to traditional combination techniques, such as score-level, feature-level and image-level techniques, we introduce an inventive fusion rule based on MLP classifier. Operating at the score-level, this methodology entails concatenating scores derived from individual models and then training the MLP classifier to compute the fitting score.
- Facial recognition does not uniformly rely only on the complete facial structure; instead, it can be reliant on specific facial components under certain conditions. From this perspective, we explore a novel S-CNN model predicated on facial regions. The fundamental concept of our proposition is based on linking the recognition of a given facial region with the recognition of the preceding facial region. This principle draws inspiration from sequential data recognition paradigms, such as text generation. In other words, the fully connected layer of the CNN that achieves optimal recognition for the initial facial region is combined with the features of the subsequent facial region and this sequence continues. Ultimately, the fully connected layers of the composite CNNs are merged and inputted into a DNN classifier to evaluate the overall system performance.

3.1 Machine Learning-based Face-recognition Approach

The facial-recognition approach proposed based on machine learning, as depicted in Figure 1, involves the following crucial steps: 1) Image preprocessing: the initial step encompasses face detection and image cropping to isolate the faces within the input images. 2) Feature extraction: in the second step, a variety of features are extracted from the preprocessed facial images. Three distinct schemes; namely, LBP, CBFD and HOG, are employed to extract different sets of features. 3) Combination model using

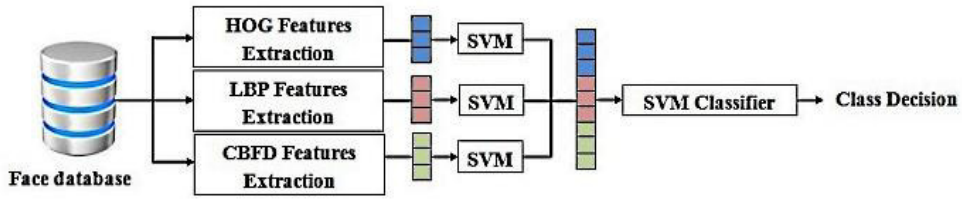


Figure 1. Fusion-based classifier combination (FCC) framework.

multiple classifiers: this step employs a combination model that integrates the outputs of three classifiers. The scores obtained from the three classifiers are combined and provided as input to the MLP classifier to determine the optimal combination for facial recognition.

3.1.1 Pre-processing Step

The preprocessing step plays a vital role in extracting valuable information from digital facial images, leading to a significant improvement in the accuracy of our facial-recognition schemes. Within our proposed schemes, this preprocessing step encompasses two primary stages: face-detection stage and cropping and resizing stage.

1) **Face detection:** In face-related applications, face detection plays a fundamental role. It involves the utilization of algorithms designed to detect and precisely locate essential points on a face, known as landmarks. The primary objective of this step is to accurately identify the facial region in order to extract features exclusively from the pertinent areas of the input image. For the purpose of facial region detection in each image, we employ a landmark-detection algorithm. Specifically, we utilize a 68-landmark shape detector that automatically identifies the facial landmark points [24]. In the first step, we focus on selecting two specific points: the top of the eyebrows and the cheeks. These points allow us to precisely localize the facial region, which is utilized in the proposed classifier combination scheme.
2) **Cropping and Resizing:** The 68-landmarks algorithm is applied to capture the distinctive characteristics associated with the facial-recognition process, such as the corners of the eyes, mouth and nose, as well as the cheeks, chin, top of the nose and forehead. These skin areas are known to be highly correlated with aging. In our study, we utilize two specific landmarks located at the top of the eyebrows and the mouth as reference points for determining the width and height of a rectangle. To ensure accurate localization of the facial region, we prefer defining slightly larger rectangles that overlap with each other. This approach enables us to cover a wider area of the face. Once the facial region is defined, the input image is cropped to include only the portion covered by the landmark-defined rectangles.

3.1.2 Feature Extraction

Feature extraction is an essential component of pattern-recognition applications, as the quality of classification results relies heavily on the distinctiveness and variability of the extracted features used to differentiate between different patterns. In our proposed methods, we employ various techniques for feature extraction, including the LBP handcraft technique, the learned handcraft technique CBFD and HOG in each method. While the LBP and HOG descriptors are explained in detail in references [21]-[22], we will provide a brief overview of the CBFD feature-extraction technique in this section.

CBFD (Compact Binary Facial Descriptors)

Jiwen et al. [23] introduced a novel feature-extraction method called the Compact Binary Face Descriptor (CBFD), which aims to enhance the performance of binary codes through a learning phase. This approach incorporates intelligence to overcome limitations and improve effectiveness. The training phase and image-feature extraction can be summarized as follows:

1) ***Training Phase***: This method focuses on the robustness of binary codes in relation to local changes in image texture. The compact binary codes are learned directly from raw pixels to represent the images. It is important to note that for better classification results, CBFD features should be constructed using a set of image samples that are provided within the same context. During CBFD feature learning, the training vectors are generated by considering the relationship between each pixel and its surrounding neighborhood. Specifically, an analysis is performed on the image using a rectangular window of size $(2R+1)\times(2R+1)$, where R is a positive integer. This window is centered on each pixel, allowing for the extraction of relevant information from the local context surrounding that pixel.

Let's define $X = [x_1, x_2, ..., x_n]$ as the set of training vectors, referred to as Pixel Difference Vectors (PDVs). These PDVs are obtained by measuring the difference between the central pixel and its neighboring pixels within a predefined window. The size of each vector is $(2R+1)\times(2R+1)-1$,

excluding the PDV between the central pixel and itself (PVD_0=0).

The goal of the CBFD feature extraction is to learn K hash functions $(w_k)_{K=1...n}$ that quantize each vector $x_n, (n = 1,...,N)$ into a binary vector $b_n = [b_{n1}, b_{n2}, ..., b_{nk}]$. This quantization is achieved through the following formula:

$$b_{nk} = 0.5 \times (sgn(w_k^T \times x_n) + 1) \quad (1)$$

Here, $sgn(v) = 1 \; if \; v \succ \tau \; and \; -1$ otherwise, where τ denotes the threshold used for binary conversion of features.

To build the projection matrix w, which comprises all the hash functions w_k, we initialize it with the K first eigenvectors of the covariance matrix $(C = XX^T)$. Then, an optimization task is performed to minimize the objective function, $J(w_k)$ defined as:

$$min J(w_k) = J_1(w_k) + \lambda_1 \times J_2(w_k) + \lambda_2 \times J_3(w_k) \quad (2)$$

The parameters λ_1 and λ_2 are predefined and used to balance the effects of different terms. The terms J_1, J_2 and J_3 are selected to ensure that: (1) The variance of the learned binary codes is maximized; (2) The quantization loss between the original feature and the encoded binary codes is minimized; (3) The feature bins in the learned binary codes are evenly distributed as much as possible.

Codebook Learning: The purpose of the codebook is to reduce the number of binary vectors associated with each image. The training vectors (PDVs) are projected onto the matrix w and then, the k-means clustering algorithm is applied to obtain the centroids of the resulting binary vectors. These centroids form the codebook, which represents the classes.

2) ***Image-feature Extraction Phase***: The feature-extraction process relies on the projection matrix w (CBFD feature) and the codebook obtained during the training phase. After obtaining all the PDV vectors $X = [x_1, x_2, ..., x_n]$ for the image, their binary counterparts are determined by projecting them onto the matrix w:

$$V_b = 0.5 * \left(sgn\left(w^T * X\right) + 1\right) \quad (3)$$

Each binary vector is then replaced with the closest vector coordinate in the codebook (bin). Subsequently, a histogram is constructed using the different coordinates, representing the entire image feature. To extract discriminative feature vectors, the raw image is segmented into multiple regions, treating each region as an individual image with its own CBFD features (w) and codebook. For each region, a histogram (H_s) is created. Finally, concatenating all the histograms results in a comprehensive vector (v) that represents the entire image:

$$v = [H_1, H_2, ..., H_M] \quad (4)$$

Where, M represents the number of regions. In our experimental results, our primary objective is to determine the number of regions that yields the highest accuracy for the facial age-estimation system. We systematically vary the number of regions to evaluate its impact on the performance of the system, aiming to identify the optimal configuration that maximizes accuracy in estimating facial age.

3.1.3 Fusion-based Classifier Combination (FCC)

Combining the decisions of multiple classifiers is an effective approach for improving classification rates, particularly in challenging pattern recognition problems. Extensive research has shown that, in many applications, fusing the outputs of multiple simpler classifiers tends to yield better recognition rates compared to relying on a single, more complex classifier. This fusion of multiple classifiers leverages their individual strengths and can lead to enhanced performance in recognizing and categorizing patterns accurately [25].

This research introduces a novel model for combining classifiers: Fusion-based Classifier Combination (FCC). The FCC method assumes that all classifiers are trained using the entire feature space and are both competitive and complementary to each other. It combines the output scores of all classifiers to make a final decision, leveraging their collective knowledge and capabilities. The proposed combination model can be summarized as follows:

- First, we have a training sample set consisting of pairs (x_i, y_j), where *i* ranges from 1 to *n*. Each sample x_i is described by d-dimensional features in a feature space $(x_i \in R^d)$, while y_i represents the corresponding category label of the sample, taking values from the set {1, n}. The number of dimensions in the feature space is denoted by *d*.
- Next, each basic classifier *j* receives a set of input data and makes predictions for each input, resulting in a score vector of size *n* representing the probabilities assigned to each class:

$$[S_j^1, S_j^2, ..., S_j^n]^T \in [0;1] \tag{5}$$

Considering the scores assigned to class *i* by base classifier *j* as S_j^i, a Multilayer Perceptron (MLP) network is employed. This MLP network consists of an input layer that takes in the obtained scores, a single hidden layer and utilizes the sigmoid activation function. The combination model makes a decision for class *i* based on the output layer of the MLP classifier, which is determined by the following formula:

$$C = \sum_{i}^{n} w_{ij} \times S_j^i + b_j \tag{6}$$

Here, w_{ij} represents the weights and b_j is the bias value of classifier *j*.

- Additionally, let $v = (v_1, v_1, ..., v_m)$ denote the actual output vector of the model, where the components $v_i \ (for\ i = 1,...,M)$ represent the combination classifier's final determination of the probability of the input samples belonging to class C_i. To update the weights and bias value, it is necessary to compute the prediction error of the model. This can be achieved by using Formula (7) to calculate the error of the j^{th} node in the output layer. The prediction error ε of the j^{th} node in the output layer is given by:

$$\varepsilon = v_j(1 - v_j) - (t_j - v_j) \tag{7}$$

Here, t_j represents the desired output value of the model.

3.2 S-CNN Deep Learning-based Face-recognition Approach

When a subject is asked to confirm the relationship between two face images, it is likely that his/her attention will be focused on specific facial features, such as the eyes, mouth and nose. We believe that these facial key-points are crucial for facial-recognition analysis. Furthermore, geometrically, there exists a high relationship between the different regions within face images. Let's consider the baseline formed by connecting the centers of the two eyes. Assuming the distance between the eye centers is represented by d, the vertical distances from the nose, eyebrows and mouth to this baseline offer valuable information for distinguishing between faces.

Our proposal introduces an innovative approach called "sequential facial region-based face recognition" aimed at improving the performance of facial-recognition systems. This novel approach treats facial images as a sequence of data, drawing inspiration from the progress made in tasks involving sequences, such as text and video recognition. Our methodology involves the use of multiple individual Convolutional Neural Networks (CNNs), as visually depicted in Figure 2. Each of these CNNs is purposefully designed to handle input data from a specific facial region, facilitating a thorough analysis of various facial components to enhance recognition accuracy.

Consider the representation of a facial image as a sequence of facial regions, denoted as *x*. At each discrete time step *t*, we identify a specific facial region, denoted as $x^{(t)}$, which serves as the input to the corresponding basic CNN. For each of these time steps, we compute a hidden state, $FC^{(t)}$, which plays a crucial role as the network's "memory." This hidden state is determined by combining information from the current input $x^{(t)}$ and the hidden state from the previous time step $FC^{(t-1)}$. Mathematically, this combination is achieved through concatenation and it can be expressed as shown in Equation (8).

$$CNN^{(t)} = x^{(t)} \oplus FC^{(t-1)} \tag{8}$$

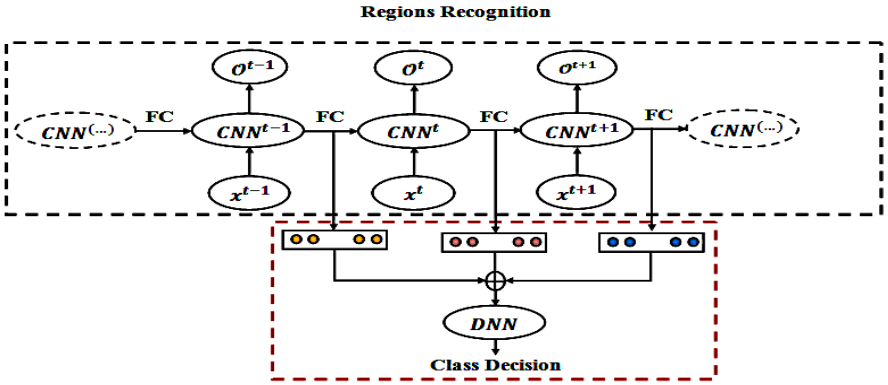


Figure 2. S-CNN deep learning-based face-recognition framework.

The result of this computation, represented as $O^{(t)}$, encapsulates the information and features that the CNN has extracted from the specific facial region $x^{(t)}$.

Moving to the second stage of our proposed approach, we consolidate the most effective fully connected layers from each of the individual CNNs. These layers have proven to be adept at identifying and processing the input facial regions optimally. This combination of fully connected layers generates a feature vector, a comprehensive representation of information from the concatenated feature vectors. This feature vector is subsequently employed as input for a Deep Neural Network (DNN) classifier. Leveraging the richness of information contained in this combined feature vector, the DNN classifier ensures efficient and effective face recognition.

4. Experiments and Discussion

A series of experiments were conducted to validate the effectiveness of the proposed methods. Each experiment focused on evaluating and comparing the performance of these methods using three well-known datasets: LFW [26] orL [27] and IJB-C [28]. These datasets were chosen, as they provide a diverse range of face images and serve as common benchmarks in the field of facial recognition. The experiments aimed to assess the accuracy and robustness of the proposed methods on these datasets, providing empirical evidence of their effectiveness in real-world scenarios.

4.1 Datasets' Description

To assess the effectiveness of the proposed methods, we conducted evaluations using three facial datasets: LFW, ORL and IJB-C.

4.1.1 LFW Dataset

The LFW dataset contains 5,749 unique individuals. Among these individuals, 1,680 have multiple images stored in the dataset, while the remaining 4,069 have only a single image. These images are saved as JPEG files and have dimensions of 250 by 250 pixels. The majority of the images are in color, although a small portion of them are grayscale. To obtain these images, the 68-landmark shape detector [24] is utilized, which accurately identifies the location of 68 facial landmarks. Subsequently, the detected faces undergo a process of resizing and cropping to achieve a consistent and fixed size.

4.1.2 ORL Dataset

On the other hand, the ORL dataset is a well-established dataset extensively employed in face-recognition research. It comprises a set of grayscale face images obtained from 40 distinct individuals. Each individual contributes ten images to the dataset. The images in the ORL dataset have a resolution of 92 by 112 pixels and are stored in a standard JPEG format. The images are captured under controlled conditions, incorporating variations in facial expressions, lighting conditions and slight pose changes. The subjects in this dataset encompass diverse genders, ages and ethnicities, making it a suitable resource for assessing the performance of face-recognition algorithms across a broad range of individuals. The dataset is commonly used for tasks, such as face detection, face recognition and facial-expression analysis.

4.1.3 IJB-C Dataset

The IJB-C (IARPA Janus Benchmark-C) dataset is widely used in the field of face recognition. It is designed to evaluate and advance the performance of face-recognition algorithms under challenging real-world conditions. The dataset contains a total of 31,334 still images, with 21,294 images featuring human faces and 10,040 images containing non-face content. On average, there are approximately 6 images available for each subject in the dataset. These images capture various facial expressions, poses and lighting conditions, making it a diverse and challenging dataset for face-recognition tasks. In addition to still images, IJB-C includes 117,542 frames extracted from 11,779 full-motion videos. Each video typically contains multiple frames of the same subjects, contributing to a more comprehensive evaluation of face-recognition algorithms. IJB-C is accompanied by a well-defined evaluation protocol that specifies how to split the dataset into training and testing sets, as well as the performance metrics used to assess face-recognition algorithms.

4.2 Protocol Description

In our experimental setup, we partitioned the face images from the ORL dataset into two distinct sets. The training samples for face-recognition systems consisted of 240 face images, comprising 6 images from each subject. The remaining 160 face images from the ORL dataset were reserved for testing purposes. Additionally, for the LFW dataset, we utilized 3300 face images (equivalent to 80% of the dataset) as the training samples for face recognition systems, while the remaining 769 face images from the LFW dataset were allocated for testing.

IJB-C introduces a comprehensive evaluation framework comprising eight distinct protocols for assessing the performance of face detection, verification, recognition and clustering across different scales and scenarios. In our study, we have specifically focused on the 1: N mixed recognition protocol, which assesses algorithms' capabilities in identification scenarios. Within this framework, there are two separate galleries; namely, Gallery 1 (referred to as G1) and Gallery 2 (referred to as G2). Each gallery contains one template per subject, which is generated by randomly selecting a half of the subject's still images. The remaining media instances are allocated to the probe set. G1 encompasses 1,772 subjects, accompanied by 5,588 still images, while G2 comprises 1,759 subjects and 6,011 still images. It's important to note that these galleries are entirely distinct from each other, facilitating open-set identification scenarios.

4.3 Evaluation Metrics

In order to assess the effectiveness of our newly proposed face-identification system, we performed thorough evaluations on a range of datasets, which encompassed ORL, LFW and IJB-C. These evaluations were carried out using the accuracy evaluation metric. Accuracy serves as a fundamental and extensively employed measure in classification systems, particularly in facial recognition. It determines the overall precision of the model by quantifying the ratio of correct predictions to the total predictions made. Mathematically, accuracy is defined as follows:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \times 100\% \quad (9)$$

In the context of facial-recognition systems, a prediction is considered accurate if the system successfully identifies or verifies the individual in the image. Conversely, an inaccurate prediction occurs when the system fails to recognize the individual or incorrectly identifies them as someone else. Accuracy serves as an easily comprehensible metric that offers a broad assessment of the system's effectiveness.

4.4 Evaluation of Performance

The experimental results in this section are divided into two main parts. The first part examines the outcomes of the FCC approach that was proposed. The second part presents the experimental results of the S-CNN approach. Following these analyses, a comparison is conducted among the leading systems to determine the most effective one.

4.4.1 Parameters' Setting

A parameter-setting phase is conducted for the feature-extraction algorithms used in the proposed FCC approach; namely, LBP, CBFD and HOG, before evaluating their performance.

1) LBP Parameters' Setting

In order to enhance the performance of the LBP feature descriptors, two key parameters are considered: the radius r of the pattern surrounding the central pixel and the number of points along the outer radius p [21]. To determine the optimal values for these parameters, a series of experiments were conducted. The LBP algorithm was tested with different combinations of samples and radius values, such as (12, 2), (12, 4), (12, 6), (12, 8), (16, 2), (16, 4), (16, 6) and (16, 8). The objective behind this parameter variation was to achieve improved results and enhance precision in the LBP algorithm when used in conjunction with an SVM (Support Vector Machine) classifier. Table 1 demonstrates the performance of the proposed system, with variations in both the number of samples and the radius *(p,r)*, while employing the SVM classifier.

In the context of the ORL dataset, the table reveals that the LBP descriptor achieves the highest level of accuracy for face recognition. More precisely, when employing **16** samples and a radius of **4**, the LBP descriptor achieves an impressive accuracy precision of **85.1%**. This underscores the exceptional performance of the LBP descriptor when it comes to recognizing faces in the ORL dataset. Turning our attention to the LFW dataset, the table indicates a respectable face-recognition accuracy of **76.8%**, even if lower than what was achieved in the ORL dataset. Nevertheless, the LBP descriptor remains effective in recognizing faces within the LFW Dataset, albeit with a slightly reduced level of accuracy compared to its performance in the ORL dataset. In summary, within the IJB-C dataset, we observe a slight reduction in face-recognition accuracy, specifically reaching **73.4%**, when using a configuration of **12** samples and a radius of **2**. This decline can be attributed to the presence of lower-quality images within the validation protocol of IJB-C.

Table 1. Improving accuracy of SVM using LBP-based features.

Datasets	LBP Parameters (*p, r*)	Accuracy [%]
ORL	(16, 4)	**85.1**
LFW	(12, 8)	**76.8**
IJB-C	(12, 2)	**73.4**

2) HOG Parameters' Setting

Similarly, a parameter-setting phase is conducted for the HOG algorithm. This phase focuses on two important parameters: the size of the blocks and the percentage of overlap between adjacent blocks. Specifically, we are interested in determining the optimal block size, while studies suggest that a 50% overlap between blocks is sufficient for effective algorithm performance [22]. To obtain better parameters, we conducted experiments by testing the HOG algorithm with blocks of different sizes. For example, we examined block sizes ranging from 10×10, 12×12 and so on, up to 32×32, while maintaining the same percentage of overlap. Varying the block size enables us to assess and identify the optimal configuration that yields improved results in terms of precision. Table 2 presents the best results obtained with the ORL, LFW and IJB-C datasets using different block sizes. This table highlights the impact of varying block sizes on the performance of the HOG algorithm and provides insights into the effectiveness of different configurations for face recognition.

From the results presented in Table 2, it is evident that face recognition achieves higher levels of accuracy in the context of the ORL dataset. To provide more detail, when utilizing a block size of **16×16**, the SVM algorithm, driven by the HOG technique, attains an impressive accuracy precision of **87.0%**. This result underscores the HOG algorithm's effectiveness in accurately identifying faces within the ORL dataset. Similarly, in the case of the LFW dataset, the recognition rates obtained exhibit competitive performance when juxtaposed with those observed in the ORL dataset. An accuracy rate of **79.8%** is achieved with a block size of **20×20**, indicating the robust performance of the HOG feature-extraction method in recognizing relationships within the LFW dataset. These outcomes closely mirror the results obtained in the ORL dataset. However, when turning our attention to the IJB-C dataset, we note a slight decrease in face-recognition accuracy, particularly at **76.2%**, when employing a block size of **12×12**. This decrement can be attributed to the dataset's inclusive nature, encompassing various subject categories and factors, such as facial hair, skin color and substantial pose variations.

Table 2. Improving accuracy of SVM using **HOG**-based features.

Datasets	HOG Parameters (*w*×*w*)	Accuracy [%]
ORL	16×16	**87.0**
LFW	20×20	**79.8**
IJB-C	12×12	**76.2**

3) CBFD Parameters' Setting

The CBFD feature-learning technique employs a predefined set of parameters tailored to our specific requirements. However, certain parameters require testing and fine-tuning to optimize the performance of our age-estimation system using this feature-extraction method. Through a series of experiments, we aimed to identify the most suitable parameters for CBFD in order to enhance our system's performance.

The CBFD algorithm relies on key parameters, including the window size, binary threshold, quantization method and projection matrix size. In our experimentation, we specifically focused on investigating the impact of the window size parameter. Our goal was to determine the optimal window size that would result in enhanced performance based on the metrics that we considered. To accomplish this, we conducted experiments using various combinations of region sizes including (3, 3), (5, 5), (7, 7), (9, 9), (11, 11), (13, 13), (15, 15) and (17, 17). By varying this parameter, we aimed to find the window size that yielded the best performance based on the metrics that we considered. Additionally, for the quantization parameter, we utilized the adaptive-quantization method with a defined threshold of **0.9**. We chose this specific approach to discretize the continuous-valued features in the CBFD algorithm. Moreover, for the feature normalization parameter, we employed Z-score normalization, which helps standardize the input data. This normalization technique ensures that the features are invariant to variations in image appearance and illumination. Furthermore, we utilized L1 regularization with a lambda value of **0.01** as a parameter to prevent overfitting and promote generalization in the CBFD algorithm.

Table 3 shows the results obtained from facial-recognition experiments conducted using the CBFD technique, wherein different window size parameters were employed. The findings demonstrate that when a window size of 7×7 is used, an impressive accuracy of **88.9%** is attained when applied to the ORL dataset. Likewise, with the LFW dataset, employing a window size of 13×13 yields a recognition rate of approximately **81.6%**. However, upon examining the IJB-C dataset, a minor decline in face-recognition accuracy is observed, specifically registering **79.54%** when a window size of 17×17 is utilized.

Table 3. Improving accuracy of SVM using **CBFD**-based features.

Datasets	Window size (*n* ,*n*)	Accuracy [%]
ORL	7×7	**88,9**
LFW	13×13	**81,6**
IJB-C	17×17	**79.54**

4.4.2 Fusion-based Classifier Combination (FCC) Performance

In this step, the research study utilizes the optimal parameters obtained from each feature-extraction technique to generate recognition scores. These scores are then concatenated and utilized as inputs for the multilayer perceptron (MLP) classifier, which serves as the combination model. Initially, support vector machine (SVM) classifiers are trained using two different kernel methods for both the LFW and ORL datasets. The selected parameters for SVM training are n = 3 for the polynomial kernel and σ = 0.125 for the RBF kernel. A value of C = 0.2 is employed during SVM training. Subsequently, the MLP classifier, acting as the combination model, is trained using the ReLU activation function in the hidden layer to introduce nonlinearity and the Softmax activation function is utilized in the output layer.

Table 4 provides a comprehensive overview of the performance of the MLP fusion technique when applied to three distinct datasets: LFW, ORL and IJB-C, utilizing different kernel methods. Notably, the RBF kernel emerges as the top performer across all three datasets, achieving the highest levels of accuracy. Specifically, when employing the RBF kernel, it achieves remarkable accuracy rates of

98.48% for the ORL dataset, **82.43%** for the LFW dataset and **81.84%** for the IJB-C dataset. These results underscore the robust accuracy levels that each kernel method can achieve when tailored to the specific characteristics of the respective datasets.

Table 4. Analyzing the statistical properties of SVM fusion with kernel methods.

Faces datasets	Kernel method	Accuracy
LFW	Polynomial kernel	78.67%
	RBF kernel	**82.43%**
ORL	Polynomial kernel	93.72%
	RBF kernel	**98.48%**
IJB-C	Polynomial kernel	75.45%
	RBF kernel	**81.84%**

The RBF kernel, in particular, stands out as the preeminent choice, demonstrating the highest recognition accuracy among the two kernel methods examined. Its superior performance makes it a widely preferred approach in fusion problems. Furthermore, it offers the practical advantage of requiring fewer parameters and encountering fewer numerical challenges compared to the polynomial kernel, enhancing its appeal in real-world applications.

4.4.3 S-CNN Deep learning-based Face-recognition Performance

The proposed S-CNN architecture for facial recognition incorporates a total of seven CNNs dedicated to recognizing specific facial regions (eyes, nose, mouth, top-left corner, top-right corner, bottom-left corner and bottom-right corner), as depicted in Figure 3.

Figure 3. The seven face regions used in the proposed S-CNN: the cropped image and its local regions, including the mouth, nose, eyes, top-left Corner, top-right corner, bottom-left corner and bottom-right corner.

In order to achieve superior accuracy in facial recognition, a series of experiments were conducted for each model, focusing on each individual facial region. These experiments aimed to identify the optimal parameters for the CNNs. The parameters considered include the choice of filter sizes (3×3, 5×5, 7×7 and 11×11) and the number of filters (8, 16, 32, 64 and 128) in the initial layer. This parameter-selection process was replicated for the subsequent layers within the architecture. Due to the comprehensive nature of the results obtained (resulting in 7×4×4=112 possibilities), Table 5 provides a concise summary of the highest accuracies achieved using the LFW, ORL and IJB-C datasets, with the selection of filter sizes and numbers serving as the key determining factors.

Table 5. Effectiveness of CNNs for facial-regions recognition.

	Datasets	LFW	ORL	IJB-C
Facial Regions	**Eyes**	73.74 %	94.23 %	**82.85%**
	Nose	71.80 %	92.33%	79.54%
	Mouth	72.10%	96.37%	77.60%
	Top-left corner	74.10 %	95.28%	75.76%
	Top-right corner	83.90%	**97.95%**	74.88%
	Bottom-left corner	84.20 %	96.10%	77.11%
	Bottom-right corner	**85.60 %**	97.49%	79.19%

The results presented in Table 5 clearly demonstrate the effectiveness of our proposed facial-recognition method in identifying facial regions across different datasets. In the case of the LFW

dataset, we observe satisfactory performance, with the bottom-right corner particularly noteworthy, achieving an impressive accuracy **of 85.60%**. When applied to the ORL dataset, our method excels even further, achieving higher accuracy rates. Specifically, the top-right corner stands out with exceptional accuracy, reaching an impressive accuracy of **97.95%**. In the context of the IJB-C results, we witness significant improvements in performance compared to the FCC approach, with our method achieving an impressive accuracy of **82.85%**.

In order to achieve accurate face recognition and effectively handle variations, the fully connected layers of the basic CNN are concatenated to form the final feature vector. This feature vector is then utilized as input for the DNN classifier, enabling robust and precise face recognition.

In our experiments, we illustrate our process of determining the ideal number of neurons for the hidden layer as shown in Figure 4. We conducted a series of tests using MLP classifier with varying numbers of neurons in the hidden layer, ranging from 10 to 100. We maintained a consistent number of maximum iterations (2000 to 5000) and employed mean squared error (MSE) training. The transfer functions utilized were sigmoid functions.

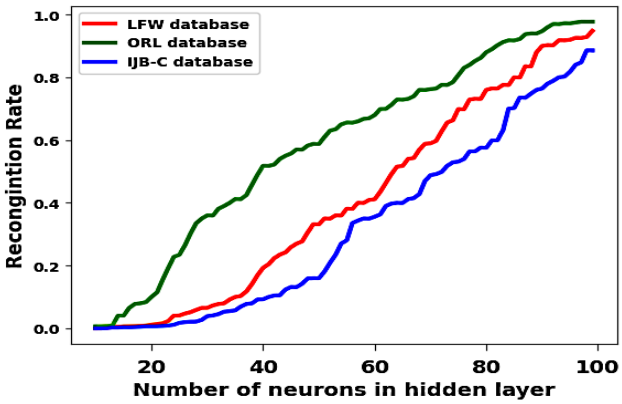


Figure 4. Recognition rates with the proposed S-CNN for LFW ORL and IJB-C dataset.

Figure 4 illustrates the performance evaluation conducted on the proposed facial-recognition system. The key observations and insights derived from this figure analysis can be outlined as follows:

- **High Recognition Rates with Over 90 Neurons:** One of the significant findings is the substantial improvement in recognition rates when employing more than 90 neurons in the hidden layer of the DNN classifier. In this configuration, the recognition rates consistently reached impressively high levels, ranging from **95.54%** to **97.75%**. This result underscores the effectiveness of the DNN architecture when coupled with CNNs for facial-recognition tasks.

- **Dataset-specific Variation:** The figure underscores the significance of dataset selection in influencing recognition performance. Notably, the recognition results for the ORL dataset outperformed those for the LFW and IJB-C datasets. Specifically, the ORL dataset achieved a recognition rate of **97.75%**, while the LFW and IJB-C datasets achieved recognition rates of **92.90%** and **88.59%**, respectively.

- **Challenges with 20-60 Neurons:** An intriguing observation pertains to the use of a relatively small number of neurons, specifically in the range of **20** to **60** neurons, within the hidden layer. During this range, the MLP algorithm, a component of the DNN, encountered convergence issues when applied to the LFW and IJB-C datasets. This issue can be attributed to the insufficient capacity of the hidden layer to effectively train the MLP classifier, highlighting the sensitivity of model architecture to dataset characteristics.

- **Dataset-specific Challenges:** The figure elucidates the specific challenges posed by the LFW and IJB-C datasets. The LFW dataset's difficulties are attributed to the considerable variations in facial orientation and expressions, which can complicate the recognition process. In contrast, the IJB-C dataset exhibits variations in both height and low image quality of the facial data, further complicating accurate recognition.

4.5 Evaluation of the Proposed S-CNN Model on SoTA Loss Functions

The central aim of face recognition, which includes both face verification and identification, is centered on the differentiation of facial features. However, the conventional Softmax loss function utilized in deep Convolutional Neural Networks (CNNs) often proves inadequate in terms of its discriminative capacity. To address this limitation, a variety of novel loss functions emerged in recent times, including Large Margin Cosine loss (CosFace) [29], Additive Angular Margin Loss (ArcFace) [30] and SphereFace Loss [31].

These advanced loss functions are designed to enhance the discriminative power of neural network feature embeddings by promoting a specific relationship between feature vectors and class centroids.

In this section, we assess the performance of the proposed S-CNN model by integrating, separately, two loss functions: CosFace and ArcFace into our proposed S-CNN model.

Implementation Details

The key steps to integrate each loss function (CosFace and ArcFace) into a basic CNN are:

1. In the last fully connected layer of each CNN in our model, we incorporated an additional layer designed for the computation of the integrated loss function. This layer accepts the feature vectors produced by the preceding layers as input and computes the specified loss function.

The employed loss functions can be expressed in the following manner:

- **CosFace Los Function**

$$L_A = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{(cos(\theta_{yi})-m)}}{e^{(cos(\theta_{yi})-m)} + \sum_{j=1, j\neq y_i}^{N} e^{cos\,\theta_j}} \tag{10}$$

The key parameters used in the CosFace loss are: 1) the parameter *s* which controls the scaling of the cosine similarity scores. It determines how much we want to magnify or shrink the angular margin applied to the cosine similarity values and 2) the parameter *m* which specifies the angular margin added to the cosine similarity between the features and the weight vectors associated with the correct classes.

- **ArcFace Los Function**

$$L = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\cos(\theta_{yi}+m)}}{e^{\cos(\theta_{yi}+m)} + \sum_{j=1, j\neq y_i}^{N} e^{\cos\,\theta_j}} \tag{11}$$

In this context, N denotes the batch size, y_i corresponds to the true label of the i^{th} example and *m* stands for the angular margin. The primary objective of the loss function is to optimize and increase the angular separation between the correct class and all other classes.

2. The output produced by the specified loss layer served as the ultimate output of our CNN model. This output was subsequently employed for both the training and evaluation or test stages of our experiments.
3. Finally, we evaluate our trained model on test datasets, taking into account the specified loss for feature embedding and classification.

In our experimental setup, we selected the margin values of $m = 0.35$ for CosFace and $m = 0.50$ for ArcFace. These choices were made based on their proven effectiveness in achieving strong performance, especially on low datasets, as demonstrated in previous research [30].

Table 6. Performance analysis of the proposed S-CNN method on CosFace and ArcFace loss functions.

S-CNN-based loss function	ORL	LFW	IJB-C
S-CNN based Softmax	**97.75%**	**92.20%**	**88.59%**
S-CNN based CosFace	**97.96%**	**93.35%**	**91.60%**
S-CNN based ArcFace	**98.95%**	**96.80%**	**92.25%**

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

As depicted in Table 6, when evaluating the performance of the proposed S-CNN method on the ORL, LFW and IJB-C datasets, the S-CNN model employing the ArcFace loss function achieved the highest level of accuracy, outperforming both the Softmax and CosFace variants. Specifically, it attained recognition rates of **98.95%** on the ORL dataset, **96.80%** on the LFW dataset and **92.25%** on the IJB-C dataset. The S-CNN approach using the CosFace loss also demonstrated strong performance, surpassing the Softmax variant on both datasets, with recognition rates of **97.96%** for ORL, **93.35%** for LFW and **91.60%** for IJB-C. In contrast, the S-CNN method employing the Softmax loss achieved the lowest recognition rates among the three methodologies, with rates of **97.75%** for ORL, **92.20%** for LFW and **88.59%** for IJB-C. Additionally, it is noteworthy that the ORL and LFW datasets consistently yielded higher recognition rates compared to the IJB-C dataset across all three loss functions, indicating variations in dataset characteristics and the effectiveness of the model.

5. COMPARISON STUDY

In this section, we conduct a comprehensive performance comparison of our proposed S-CNN method with recent state-of-the-art techniques [10]-[12], [16]-[18], including those based on various loss functions [9], [13]. Table 7 displays the accuracy results of these methods, covering both machine-learning and deep-learning approaches.

Regarding our machine learning-based method, our evaluation reveals that the fusion of classifiers achieves an impressive accuracy of **98.48%** on the ORL dataset, **82.43%** on the LFW dataset and **81.84%** on the IJB-C dataset. Notably, our proposed method outperforms the approach introduced by Muqeet et al. [11], which achieved an accuracy of 97.00%. Furthermore, when compared to deep learning-based methods, our approach surpasses Kong et al.'s [18] results using PCANet + KNN and PCANet + SVM, achieving accuracies of 91.50% and 97.50%, respectively.

Additionally, Table 7 provides a detailed comparison of the performance of our proposed deep learning-based method against recent deep-learning approaches. On the ORL dataset, our method attains an accuracy of **97.75%**; outperforming Kong et al.'s [18] results with accuracies of **91.50%** and **97.50%**. For the LFW dataset, our approach achieves an accuracy of approximately **92.20%**, surpassing the combination of FaceNet + RF [10] with an accuracy of **89.10%** and the combination of MLP + MFM with CNN [16] with an accuracy of 84.5%. Furthermore, our proposed method competes closely with Storey et al. [17] method, which achieved an accuracy of **93.60%**.

Table 7. Comparative analysis of the proposed methods with state-of-the-art (DL and ML techniques).

Machine learning

	Datasets		
Method	**High Quality**		**Mixed Quality**
	ORL	**LFW**	**IJB-C**
KNN (DWT+LBP) [11]	97.00%	-	-
Proposed method	**98.48%**	**82.43%**	**81.84%**

Deep learning

FaceNet + RF [10]	-	89.10%	-
LBP + Ensemble CNN [12]	100%		-
MLP + MFM in CNN [16]	-	84.50%	-
3D-CNN+ResNe [17]	-	93.60%	-
PCANet + KNN [18]	91.50%	-	-
PCANet + SVM[18]	-	97.50%	
ResNet-100 $_{CosFace}$ [9]	-	-	**92.20%**
ResNet-100 $_{ArcFace}$ [9]			**95.20%**
LMCL$_{CosFace}$ [13]	-	**92.69%**	-
LMCL$_{ArcFace}$ [13]		**93.30%**	
Comparison with state-of-the-art loss functions			
S-CNN based Softmax	**97.75%**	**92.20%**	**88.59%**
S-CNN based CosFace	**97.96%**	**93.35%**	**91.60%**
S-CNN based ArcFace	**98.95%**	**96.80%**	**92.25%**

Finally, we compare our proposed S-CNN model with state-of-the-art loss function methods [9], [13]. The comparison is presented in Table 7, where our model outperforms the approach introduced in [13], achieving **93.35%** and **96.80%** accuracy for the CosFace and ArcFace loss functions, respectively, on the LFW and IJB-C datasets. Compared to the method presented in [9], our proposed approach achieves competitive accuracies of **91.60%** for the CosFace loss function and **92.25%** for the ArcFace loss function, as opposed to **92.20%** for CosFace and **95.20%** for ArcFace obtained in [9].

6. CONCLUSION

This research paper addresses the challenges in facial recognition through the introduction of two innovative approaches: FCC and S-CNN. The effectiveness of three techniques; namely, LBP, HOG and CBFD, is evaluated in overcoming these challenges. The proposed solution involves the utilization of a novel multi-classifier combination model and a unique method for extracting high-level features from multiple image regions treated as sequential data using an ensemble of CNNs, followed by a DNN classifier for facial recognition.
The experimental results obtained from renowned facial datasets, including LFW, ORL and IJB-C, reveal the competitive performance of both the proposed multi-classifier combination model and the S-CNN deep-learning model when compared to state-of-the-art methods. Additionally, we have assessed the effectiveness of the proposed S-CNN model alongside state-of-the-art loss functions, such as CosFace and ArcFace. Based on the results that we have obtained from our experiments, we can illuminate specific strengths and weaknesses of our approach as follows:

- The experimental results show that FCC method based on combination at matching score level is likely to provide better recognition performance, as it contains more contented information which is both feasible and practical.
- This paper illustrates how to use CNN as a sequential model and we believe that it may open a door towards alternative to deep neural networks for many tasks. Traditional CNN will process an input and move onto the next one disregarding its sequence. In the proposed S-CNN, an image is considered as a series of sequential face regions that needs to be followed in order to understand. In other words, the first CNN receives a region of an image and passes it as a feature vector to the next CNN to predict the next face region based on the previous region and so on.
- Furthermore, we would like to mention that it is possible that the proposed S-CNN model could be used in other applications, such as age estimation, gender prediction or facial-emotion recognition.
- However, the proposed methods need to be accurate and robust enough to handle the variability and diversity of faces and datasets.
- In future research, we can explore the application of attention mechanisms to automatically identify distinguishing facial regions while effectively minimizing the impact of noisy areas.

REFERENCES

[1] I. Bendib, A. Meraoumia, M. Y. Haouam et al., "A New Cancelable Deep Biometric Feature Using Chaotic Maps," Pattern Recognition and Image Analysis, vol. 32, no. 1, pp. 109–128, 2022.

[2] M. Y. Haouam, A. Meraoumia, L. Laimeche and I. Bendib, "S-DCTNet: Security-oriented Biometric Feature Extraction Technique: An Effective Pathway to Secure and Reliable Biometric Systems," Multimedia Tools and Applications, vol. 80, pp. 36059–36091, 2021.

[3] R. Rameswari, K. S. Naveen, A. M. Abishek and C. Deepak, "Automated Access Control System Using Face Recognition," Materials Today: Proceedings, vol. 45, Part 2, pp. 1251-1256, 2021.

[4] N. Singhal, V.Ganganwar, M. Yadav, A. Chauhan, M. Jakhar and K. Sharma, "Comparative Study of Machine Learning and Deep Learning Algorithm for Face Recognition," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 07, no. 03, pp. 313-325, September 2021.

[5] Z. Wanling and X .Shijun, "Face Anti-spoofing Detection Based on DWT-LBP-DCT Features," Signal Processing: Image Communication, vol. 89, p. 115990, DOI: 10.1016/j.image.2020.115990, 2020.

[6] S. Liangliang, W. Xia and S. Yongliang, "Research on 3D Face Recognition Method Based on LBP and SVM," Optik, vol. 220, p. 165157, DOI: 10.1016/j.ijleo.2020.165157, 2020.

[7] K. Weiyi., Y. Zhisheng and L. Xuebin, "3D Face Recognition Algorithm Based on Deep Laplacian Pyramid under the Normalization of Epidemic Control," Computer Communications, vol. 199, pp. 30-41, 2023.

[8] D. Mamieva, A.B. Abdusalomov, M. Mukhiddinov and T. K. Whangbo, "Improved Face Detection Method via Learning Small Faces on Hard Images Based on a Deep Learning Approach," Sensors, vol. 23, no. 1, p. 502, DOI: 10.3390/s23010502, 2023.

[9] G.-S. J. Hsu, H. -Y. Wu, C.-H. Tsai, S. Yanushkevich and M. L. Gavrilova, "Masked Face Recognition from Synthesis to Reality," IEEE Access, vol. 10, pp. 37938-37952, 2022.

[10] A. S. Sanchez-Moreno et al., "Efficient Face Recognition System for Operating in Unconstrained Environments," Journal of Imaging, vol. 7, no. 9, p. 161, 2021.

[11] L. Zhou, H. Wang, S. Lin. et al., "Face Recognition Based on Local Binary Pattern and Improved Pairwise-constrained Multiple Metric Learning," Multimedia Tools Application, vol. 79, pp. 675–691, DOI: 10.1007/s11042-019-08157-0, 2020.

[12] J. Tang, Q. Su, B. Su, S. Fong, W. Cao and X. Gong, "Parallel Ensemble Learning of Convolutional Neural Networks and Local Binary Patterns for Face Recognition," Computer Methods and Programs in Biomedicine, vol. 197, p. 105622, DOI: 10.1016/j.cmpb.2020.105622, 2020.

[13] J. Deng, J. Guo, N. Xue and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition, pp. 4690-4699, Long Beach, USA, 2019.

[14] Z. Mortezaie and H. Hassanpour, "A Survey on Age-invariant Face Recognition Methods," Jordanian Journal of Computers and Information Technology (JJCIT), vol. 05, no. 02, pp. 87 - 96, August 2019.

[15] M.A. Muqeet and R. S. Holambe, "Local Binary Patterns Based on Directional Wavelet Transform for Expression and Pose-invariant Face Recognition," Applied Computing and Informatics, vol. 15, no. 2, pp. 163-171, 2019.

[16] H. Yu, J. Zhao and Y. Zhu, "Research on Face Recognition Method Based on Deep Learning," Proc. of the 12th Int. Congress on Image and Signal Processing, Biomedical Engineering and Informatics (CISP-BMEI), Suzhou, China, 2019, pp. 1-5, 2019.

[17] G. Storey, R. Jiang, S. Keogh, A. Bouridane and C. Li, "DPalsyNet: A Facial Palsy Grading and Motion Recognition Framework Using Fully 3D Convolutional Neural Networks," IEEE Access, vol. 7, pp. 121655-121664, 2019.

[18] J. Kong, M. Chen, M. Jiang, J. Sun and J. Hou, "Face Recognition Based on CSGF (2D) 2PCANet," IEEE Access, vol. 6, pp. 45153-45165, 2018.

[19] Z. H. Zhou and J. Feng, "Deep Forest: Towards an Alternative to Deep Neural Networks," Proc. of the 26th Int. Joint Conf. on Artificial Intel. (IJCAI-17), pp. 3553-3559, DOI: 10.24963/ijcai.2017/497, 2017.

[20] M. Ramgopal et al., "Masked Facial Recognition in Security Systems Using Transfer Learning," SN Computer Science, vol. 4, Article no. 27, DOI: 10.1007/s42979-022-01400-w, 2023.

[21] D. Samai, A. Meraoumia, H. Bendjenna and L. Laimeche, "Oriented Local Binary Pattern (LBPθ): A New Scheme for an Efficient Feature Extraction Technique," Proc. of the IEEE Int. Conf. on Mathematics and Inf. Techn. (ICMIT), DOI: 10.1109/MATHIT.2017.8259710, Adrar, Algeria, 2017.

[22] B. Berkant, "Implementation of Hog Edge Detection Algorithm Onfpga's," Procedia - Social and Behavioral Sciences, Vol. 174, pp. 1567-1575, DOI: 10.1016/j.sbspro.2015.01.806, 2015.

[23] L. Jiwen et al., "Learning Compact Binary Face Descriptor for Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), vol. 37, no. 10, pp. 2041-2056, 2015.

[24] T. Devries, K. Biswaranjan and G. W. Taylor, "Age Estimation and Face Verification across Aging Using Landmarks," Proc. of the IEEE Canadian Conf. on Computer and Robot Vision, pp. 98–103, Montreal, Canada, 2014.

[25] S. Liu., X. Li, C. Hu et al., "Spammer Detection Using Multi-classifier Information Fusion Based on Evidential Reasoning Rule," Scientific Reports, vol. 12, Article no. 12458, DOI: 10.1038/s41598-022-16576-7, 2022.

[26] G. B Huang et al., "Labeled Faces in the Wild: A Dataset for Studying Face Recognition in Unconstrained Environments," Proc. of Workshop on Faces in 'Real-Life' Images: Detection, Alignment and Recognition, [Online], Available: http://vis-www.cs.umass.edu/lfw/, 2008.

[27] F. S. Samaria and A. C. Harter, "Parameterization of a Stochastic Model for Human Face Identification," Proc. of IEEE Workshop on Applications of Computer Vision, pp. 138-142, Sarasota, USA, 1994.

[28] B. Maze, J. Adams, J. A. Duncan et al., "IARPA Janus Benchmark-C: Face Dataset and Protocol," Proc. of the 2018 IEEE Int. Conf. on Biometrics (ICB), pp. 158-165, Gold Coast, Australia, 2018.

[29] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li and W. Liu, "CosFace: Large Margin Cosine Loss for Deep Face Recognition," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, pp. 5265–5274, DOI 10.1109/CVPR.2018.00552, 2018.

[30] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj and L. Song, "SphereFace: Deep Hypersphere Embedding for Face Recognition," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, pp. 212–220, arXiv: 1704.08063, 2017.

[31] M. Kim, A. K. Jain and X. Liu, "AdaFace: Quality Adaptive Margin for Face Recognition," Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR2022), pp.18729-18738, [Online], Available: https://openaccess.thecvf.com/content/CVPR2022/papers/Kim_AdaFace_Quality_Adaptive_Margin_for_Face_Recognition_CVPR_2022_paper.pdf, 2022.

ملخص البحث:

تبحثُ هذه الورقة في مقارنةٍ شاملةٍ بين طريقتين من طُرق تشكيل تركيبات البيانات من أجل تحسين الموثوقية لأنظمة تمييز الوجوه. الطّريقةُ الأولى تسمّى تركيبة المصنِّفات القائمة على الاندماج (FCC)، ويتألّف النّموذج في هذه الطّريقة من (3) مصنِّفات يتمّ تدريب كلٍّ منها باستخدام إحدى تقنيات استخلاص السِّمات المعروفة. أما الطّريقة الثّانية فهي طريقة التّعلُّم العميق باستخدام الشّبكات العصبية الالتفافية المتعاقِبة (S-CNN)، وفيها يتمّ استخلاص عددٍ من السِّمات عالية المستوى من مناطق مختلفة في صُوَرٍ للوجوه وإدخالها على مجموعة متسلسلة من الشّبكات العصبية الالتفافية. بعد ذلك، تُجَمَّع مُخرجات تلك الشّبكات وتُدْخَل الى شبكة تعلُّم عميق عصبية (DNN) مخصَّصة لتمييز الوجوه.

وقد جرى اختبارُ النّموذجيْن المقترحَيْن في هذه الدّراسة على ثلاثٍ من مجموعات البيانات الخاصّة بصُوَر الوجوه. وقد تمّ الحصولُ على نتائج تؤكّد تنافسية الطّريقتين المستخدّمتين عند مقارنتهما بطرق تشكيل تركيبات البيانات الخاصّة بأنظمة تمييز الوجوه.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

LONGCGDROID: ANDROID MALWARE DETECTION THROUGH LONGITUDINAL STUDY FOR MACHINE LEARNING AND DEEP LEARNING

Abdelhak Mesbah[1]*, Ibtihel Baddari[1] and Mohamed Amine Riahla[2]

(Received: 30-Aug.-2023, Revised: 3-Oct.-2023 and 24-Oct.-2023, Accepted: 31-Oct.-2023)

ABSTRACT

This study aims to compare the longitudinal performance between machine-learning and deep-learning classifiers for Android malware detection, employing different levels of feature abstraction. Using a dataset of 200k Android apps labeled by date within a 10-year range (2013-2022), we propose the LongCGDroid, an image-based effective approach for Android malware detection. We use the semantic Call Graph API representation that is derived from the Control Flow Graph and Data Flow Graph to extract abstracted API calls. Thus, we evaluate the longitudinal performance of LongCGDroid against API changes. Different models are used; machine-learning models (LR, RF, KNN, SVM) and deep-learning models (CNN, RNN). Empirical experiments demonstrate a progressive decline in performance for all classifiers when evaluated on samples from later periods. However, the deep-learning CNN model under the class abstraction maintains a certain stability over time. In comparison with eight state-of-the-art approaches, LongCGDroid achieves higher accuracy.

KEYWORDS

Android security, Malware detection, Machine learning, Adjacency matrix, Longitudinal evaluation.

1. INTRODUCTION

The ever-expanding landscape of mobile applications has brought about new challenges in ensuring the security and privacy of users' devices. Among these challenges. Android malware stands out as a persistent and evolving threat, requiring robust and effective detection mechanisms. With a market share near 72% as of the first quarter of 2023 [1], the Android platform leads the mobile OS. Its popularity, accentuated by the extensive availability of diverse third-party Android app-distribution channels, makes it a prime target for malware. It is estimated that more than 5 million Android malware samples have been seen in the wild as of the first quarter of 2023 [2].

Traditional signature-based methods have proven insufficient in keeping pace with the constantly mutating nature of malware, necessitating the exploration of advanced techniques. Aware of this fact, researchers use machine-learning (ML) and deep-learning (DL) techniques to develop malware analysis and detection systems by analyzing the behavior of the apps and extracting the set of features that best describe their behavior based on dynamic and/or static features [3]. Dynamic analysis refers to a method of analyzing the behavior of apps in real time [4], while they are running in a controlled environment, known as a sandbox. This approach involves monitoring the apps' interactions with the operating system, hardware and other apps, simulating various user inputs and system events to understand how the apps behave under different conditions. Dynamic analysis is different from static analysis, inspecting the code and data of apps without executing them [5]. Instead, it involves examining the contents of the apps, which include the app's code, resources and manifest file, among others. This analysis is performed before the app is run, giving an overview of the app's structure, potential vulnerabilities and behavior without actually executing it on a device or emulator. Both analysis approaches are complementary and can be combined for Android malware detection.

Many works and tools have been developed for malware detection, involving various analysis methods and feature usages. However, most studies present performance results using

1. A. Mesbah (Corresponding Author) and I. Baddari are with Department of Computer Science, Faculty of Sciences, LIMOSE Laboratory, University M'Hamed Bougara of Boumerdes, Algeria. Email: abdelhak.mesbah@univ-boumerdes.dz
2. M. A. Riahla is with Department of Electrical Systems Engineering, Faculty of Technology, LIMOSE Laboratory, University M'Hamed Bougara of Boumerdes, Algeria.

conventional methods without considering the period or age of the test samples used during the extraction and evaluation. As apps evolve, their characteristics and behavior may change, which can affect the accuracy and sustainability of the models. A few existing works focus on characterizing the behavior of Android apps in relation to the temporal evolution of data, specifically addressing the phenomena commonly referred to as concept drift and data drift. These studies propose methodologies that enable adaptation to changes in the data and aim to mitigate the negative consequences that may arise over time. On the one hand, for security reasons, but not necessarily for Android malware detection, [6]-[8] discussed comprehensive approaches towards understanding and adapting to the rapid evolutionary dynamics of the mobile app ecosystem, mainly focusing on Android, to improve app quality, security and usability. The authors proposed a continuous ecosystem mining and characterization approaches to systematically study and understand the evolutionary dynamics of the Android ecosystem. On the other hand, different studies contributed to a deeper understanding of the intricacies involved in malware classification and the importance of adapting to the evolutionary dynamics of the Android ecosystem to improve malware-detection accuracy and reliability. In [9]-[13], the authors underscored the importance of a comprehensive approach to understanding app evolution in enhancing malware classification. As the evolution problem is multi-faceted, it encompasses the continuous adaptation of malware to evade detection, the evolution of goodware to stay secure and functional and the changes in the Android platform that might affect both malware and goodware behaviors. The studies demonstrated that a more accurate and reliable malware classification can be achieved by eliminating experimental bias, addressing platform fragmentation and analyzing malware behavior over time.

The features used for malware detection may exhibit dominance or effectiveness in certain years, but not in others, which highlights the importance of considering the temporal aspect and continuously evaluating the effectiveness of the chosen features and models longitudinally. Hence, in this study, a longitudinal performance comparison is made between machine-learning and deep-learning classifiers, through the proposed LongCGDroid, an image-based, fully automated process to detect Android malware using static and pseudo-dynamic analysis. LongCGDroid uses a classic representation of programs in program-analysis techniques; namely, the Call Graph API, which is a representation of the interactions between various functions within a software application. It is used to visualize and analyze the flow of function calls made by an Android app during its execution. This call graph is constructed based on two main steps: the extraction of the Control Flow Graph (CFG) to reflect what a program intends to behave, as well as how it behaves (i.e., possible execution paths), such that malware behavior patterns can be captured easily and the Data Flow Graph (DFG) that represents the data dependencies between several operations, in our case links between API calls. Both CFG and DFG are extracted through static and pseudo-dynamic analysis on the instruction level (i.e., smali files in the Dalvik executions). By analyzing each call graph's semantics, we construct an abstracted 2-D adjacency matrix for each app that represents the relation between the abstracted API's calls.

The contributions of this study can be summarized as follows:

- Generating a large date-labeled dataset containing 200K Android apps (100k malware and 100k goodware) with a 10-year range from 2013 to 2022; for each year, 20K Android applications are used (10k malware and 10k goodware).
- Proposing and developing LongCGDroid, a new malware-detection image-based system, using the semantic of the call graph API with different modes of abstraction. All execution paths in terms of the invoked APIs are captured through the DFG. The pseudo-dynamic analysis is applied; i.e., apps are not executed; instead, all execution paths are analyzed *via* the DFG.
- We propose the use of different API call abstractions to represent them in the adjacency matrices; i.e., abstract API calls to either the method name (e.g. *java.io.File.getPath*), the class name (e.g. *java.io.File*) or its package name (*java.io*). Abstraction provides resilience to API changes in the Android framework as classes and packages are added and removed less frequently than single-method API calls. We evaluate the classifiers with each abstraction, as the LongCGDroid can operate with each mode.
- We propose the use of an adjacency matrix to effectively engineer Android APIs for machine-learning tasks. This approach allows better feature embedding. By only using the most popular

features, the graph's size is significantly reduced. Therefore, the adjacency matrix is also greatly simplified with an inferred size for each level of abstraction: 100×100 for package abstraction, 200×200 for class abstraction and 300×300 for method abstraction, allowing a fast data-processing stage.

- We investigate the extent of performance decay over time for various machine-learning and deep-learning classifiers trained with features extracted from date-labeled goodware and malware samples. The classifiers are then tested on goodware and malware samples from a later time period, thus mimicking a true zero-day scenario that gives a more realistic view of performance than the traditional evaluation approach.
- We conduct comparative experiments with eight state-of-the-art techniques [14]-[21], which fully demonstrate the effectiveness of our approach.

The paper is organized as follows. We begin by providing a background as contextual foundation through outlining the key concepts in Section 2, followed by a discussion of the related works on malware detection in Section 3. The LongCGDroid framework and our methodology are introduced in Section 4. LongCGDroid's evaluation and comparative analyses are presented in Sections 5. Conclusions are presented in Section 6.

2. BACKGROUND

2.1 Android APK Format

Android Application Package (APK) is the file format used to distribute and install applications on Android devices [22]. It is a compressed archive file that contains all the necessary components and resources required to run an Android application. The components of an APK include the *Manifest File*, which contains essential information about the application, such as its package name, version code and permissions. Compiled code *classes.dex* is represented as Dalvik Executable (DEX) files. These files contain the bytecode generated from the application's Java source code. The Resources folder contains files, such as images, layouts and other resource files required for the user interface and functionality. Libraries are required by the application to interact with the underlying hardware or perform certain platform-specific tasks.

2.2 Static Analysis

Static analysis is one of the most commonly used techniques in malware detection, focusing on inspecting the APK without executing the code [5]. By examining the APK's structure and contents, static analysis extracts valuable insights to assess potential threats. This analysis method is advantageous, as it allows for a rapid examination of large datasets, providing a preliminary screening of applications. During static analysis, numerous static features are extracted from the APK [3]. These features encompass various characteristics of the application, enabling the classifier to distinguish between benign and malicious samples. Some common static features include permissions, API calls, string analysis and code structure. The information extracted through static analysis is then converted into a standardized set of explanatory features that are later processed by machine-learning algorithms. To carry out static analysis on APK files, specialized tools like Androguard [23] can be employed to access fields and components declared within the manifest file. It also facilitates the construction of various graph structures representing the code's execution flow. Notably, two essential graph types are Call Graphs and CFGs. Call Graphs are built by tracing call instructions in the code, while CFGs encompass conditional and loop statements (if, switch, for, while, …etc.), reflecting the code's jumps. Two main tools can be used to extract call graphs or CFGs: Androguard and Flowdroid [24].

2.3 Pseudo-dynamic Analysis

Pseudo-dynamic program-flow analysis is a technique used in software analysis to understand the potential execution paths of a program without executing it in a real runtime environment [25]. Unlike traditional dynamic analysis, which involves running the program on an actual system, pseudo-dynamic program-flow analysis performs a form of simulated execution with code instrumentation. In this approach, one can track and log the control flow as the code executes. These additional instructions act as probes or logging points, allowing researchers to observe the program's

execution path and capture critical information about the program's behavior. Pseudo-dynamic program-flow analysis offers several advantages over traditional static analysis. It enables researchers to gain insights into how the program would behave under certain conditions without the need to execute it in a real environment. Additionally, it can help identify potential security vulnerabilities, resource bottlenecks or unintended program paths. Particularly, all execution paths can be captured in terms of the operation codes, so-called opcodes.

3. RELATED WORKS

Android malware detection is a well-studied area in the information-security literature. There have been several works that focus on machine learning-based detection. Despite this, very few of them consider the investigation of long-term performance and a longitudinal resilience evaluation. In this section, we classify them into two categories: longitudinal-based and conventional-based evaluations.

3.1 Longitudinal-based Machine-learning and Deep-learning Evaluation

The majority of existing research on Android malware detection does not address longitudinal resilience or long-term performance issues. In [26], the paper presents a dedicated longitudinal study of the performance of machine-learning classifiers for Android malware detection aiming for a sustainable system. The study is undertaken using very basic features, such as API calls, permissions and intents extracted from Android apps. It takes into consideration a dataset constructed from apps first seen between 2012 and 2016. The aim is to investigate the extent of performance decay over time for various machine-learning classifiers, such as Support Vector Machines (SVMs), Naïve Bayes (NB), Random Forest (RF), Simple Logistic (SL) and Decision Tree (DT). The SL was the most resilient over time. In [14], MAMADROID is presented as an Android malware detection system that relies on app behavior. It builds a behavioral model in the form of a Markov chain from the API call sequences, which is used to extract features and perform classification. The paper includes a longitudinal evaluation of the accuracy using a dataset of 8.5K benign and 35.5K malicious apps, collected over a period of six years, first seen between 2010 and 2016. The reported F-measure is up to 99% with a progressive diminishing performance over time, reaching average F-measure values of 86% and 75% one and two years after training, respectively. Maldozer [15] is a deep learning-based framework that relies on raw API call sequences for identifying malware apps. To evaluate their tool, they constructed a dataset of 33k goodware and 38 malware apps. It is also longitudinally evaluated on apps collected from four consecutive years, 2013 to 2016. The authors achieved 96% to 99% detection accuracy. Similarly, RevealDroid [19] addresses the increasing threat of Android malware and the limitations of existing detection techniques, particularly their inability to handle certain obfuscations and scalability issues. The features selected for RevealDroid focus on categorized Android API usage, reflection-based features and features from native binaries of apps. In [16], the authors delve into the evolving nature of malware and its implications on detection strategies, particularly emphasizing the phenomenon of concept drift. Through their system named DroidEvolver, they underscore the continuous evolution of malware as a significant challenge to cybersecurity, which in turn affects the performance of machine-learning models employed in malware detection. The approach is designed to automatically and continually update itself during malware detection without human intervention or retraining with accurate labels. DroidCat [20] introduces a dynamic app classification technique for Android malware detection and categorization. The system uses dynamic features based on method calls and inter-component communication (ICC) intents. The authors assessed DroidCat's performance across the spanning of nine years, with a relatively small dataset of 34343 samples collected from 2009 to 2017. In [21], the author presents DroidSpan, a dynamic-analysis system based on a behavioral profile for Android apps. This system captures sensitive access distribution from lightweight profiling during runtime. The system's dynamic approach focuses on the extent and distribution of exercised sensitive accesses and vulnerable method-level control flows in app executions. By leveraging this dynamic behavioral profile, DroidSpan aims to offer sustainability in malware detection, especially in the context of the evolving Android platform and its applications. Through all the longitudinal experiments made by the author over the years, the system does not exceed an F1-score of 91%.

All of these existing Android malware-detection studies have employed a relatively small dataset containing a few thousand apps. Moreover, the evaluation timelines of these studies did

not extend beyond 2018. Furthermore, none of the studies provided a comparison of performance between machine- learning and deep-learning models.

3.2 Conventional-based Machine-learning and Deep-learning Evaluation

Early research on Android malware detection predominantly employed traditional machine-learning algorithms. Most machine learning-based malware techniqes still use features such as Android app permissions, API calls and control flow graphs to distinguish between benign and malicious apps.

For instance, Jung et al. [27] proposed a malware detection approach based on the frequency of API calls. The authors' main idea is based on the construction of two ranked lists of popular Android API calls: a benign API call list and a malware API call list. The RF classifier is applied to a dataset of 60,243 apps (30,159 goodware and 30,084 malware) using each list as a feature. The evaluation shows that the classifier achieves promising results. DroidSieve [17] is introduced as a malware-classification system that employs obfuscation-resilient static analysis of Android apps. The authors delve into the challenges posed by obfuscation in Android malware, discussing various obfuscation techniques and their impacts on static analysis. They propose a diverse set of features for robust classification, covering both non-obfuscated and obfuscated malware and offering a high level of accuracy and efficiency while addressing the challenges posed by obfuscation in malware analysis and detection. In [28]-[29] the authors presented frameworks that combine permission and API calls to detect malicious Android apps using machine-learning methods, extracting permissions from each app's profile information and APIs from the APK to represent API calls and validating the algorithm's performance through experiments on real-world apps with a small dataset. Similarly, [30]-[33] proposed an intelligent model for detecting malware applications using machine-learning algorithms. The frameworks are based on static malware analysis to extract features, such as permissions and API calls using multi-level feature selection algorithms.

Graph-based approaches have gained traction over the years, leveraging the structural and behavioral characteristics of Android apps to enhance detection accuracy. DroidMiner [34] is a system that uses static analysis to automatically mine malicious program logic from known Android malware and seeks out these threat-modality patterns in other unknown Android apps. They also describe a two-tiered behavioral graph model, control-flow graphs and call graphs, for characterizing Android application behavior and labeling its logical paths within known malicious apps as malicious modalities. In [35], the paper proposes a root exploit malware recognition system called DroidExec, which reduces the impact of wide variability in Android malware detection. The system uses a Bipartite Graph Conceptual Matching of graph edit distance to fold redundant function-relation graphs and conceptually cripple wide variability. DroidOL [36] and DeepCarta [37] propose frameworks that capture security-sensitive behaviors from apps in the form of structural information captured through graphs.

With the rise of deep learning, researchers started to leverage its capabilities to achieve more robust and accurate malware detection. EveDroid [38] uses event groups to describe app behaviors in event level and function clusters to represent behaviors in each event. A neural network is used to aggregate multiple events and mine the semantic relationship among them. [39] proposes a multimodal learning approach for Android malware detection using deep-learning techniques. It uses static analysis to extract permission and hardware features. In [40]-[42], the authors propose systems that use deep convolutional neural networks to learn from opcode sequences. The systems extract raw opcode sequences from decompiled Android files and train the network to effectively learn feature information and accurately detect malicious programs. R2-D2 [43] is system that converts the bytecode of *classes.dex* from APK to RGB color codes and stores them as fixed-size color images. These images are then input to the CNN for automatic feature extraction and training. Similarly, Vu et al. [18] propose AdMat, a grayscale image-based approach to treat malware detection. The authors construct an adjacency matrix for each app, serving as input images to the CNN model. DroidDivesDeep [44] proposes a method for classifying Android malware using low-level monitorable features and deep neural networks.

In our study, we aim to compare machine-learning and deep-learning classifiers by addressing the longitudinal aspect, which offers a more authentic assessment of effectiveness compared to

conventional evaluation methodologies that overlook the temporal aspect of app appearances. We evaluate our framework LongCGDroid using a large dataset of date-labeled apps with a specific focus on utilizing graph-based features' approach, where both control and data-flow graphs are considered.

4. METHODOLOGY

The proposed LongCGDroid consists of five consecutive main stages. Figure 1 shows the framework of our approach. The first stage consists of collecting and cleaning the dataset. At the second stage, static and pseudo-dynamic analyses of Android apps are done, to extract API call graph, CFGs and DFGs of the samples under inspection. Particularly, all execution paths are captured using the extracted CFGs. These CFGs are used in the pseudo-dynamic analysis of the DFGs to extract all the invoked built-in APIs. Data- flow analysis is used to track the data across apps. It relies on an underlying abstract semantics of Android apps. Data flow shows the data dependencies between functions. At the third stage, we conduct a frequency analysis to select the APIs which are the most used. We further refine the API list to include only those with a usage difference higher than or equal to a certain threshold. Then, considering the abstraction mode, we select the number of APIs to use in order to construct and encode the adjacency matrices. At the fourth stage, we train the selected models using a part of the dataset. And finally, a longitudinal evaluation is made over years using the rest of the dataset.

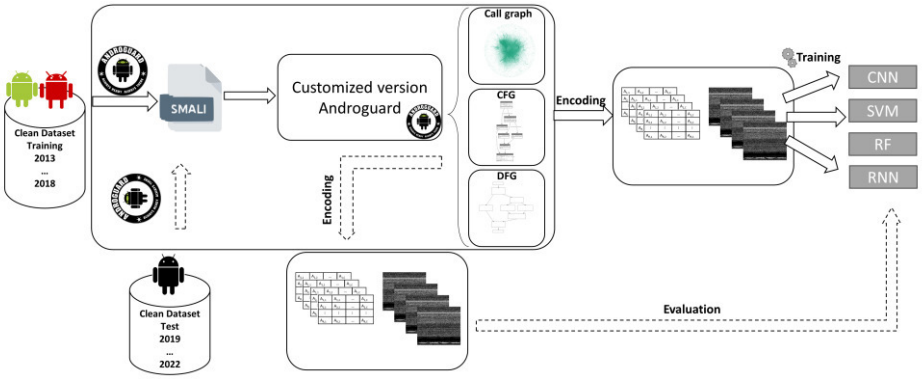


Figure 1. Process flow of LongCGDroid.

4.1 Dataset

We collect samples mainly from two datasets: Androzoo [45] for goodware apps and Virushare [46] for malware apps, as each app in these datasets is associated with the first seen timestamp, which is essential for our study. Our dataset is constructed from apps first seen between 2013 and 2022. During the collection process, we also took into consideration the apps from Maldozer [15] where we collected all the apps based on their hashes.

Initially, before cleaning, the dataset was constructed from 269970 apps; for each year, more than 20k apps are collected (> 10k goodware apps, >10k malware apps). To construct the final dataset, we considered cleaning phases (Figure 2). The first phase checks if the app is not corrupted and can be decompiled through Androguard, as some Android apps failed to decompress because of bad CRC-32 redundancy checks and errors during unpacking. Next, the app is automatically uploaded through our Python scrapper to VirusTotal [47], which is a tool that allows users to upload files, including APKs and scan them using a collection of antivirus engines. Once the app is scanned, it is labeled depending on the number of positive (malware) alerts. This number is what we call VTD (VirusTotal Detection) and it is used to relabel the samples using a threshold to establish the level of consensus required to label an APK as malware or goodware [48]. In our case, to label apps as malware, the VTD must be at least equal to 4. On the other hand, apps are labeled as goodware if the VTD is equal to 0, which means that all antiviruses marked it as clean. These verifications and labeling choices are influenced by the nature of the dataset, which includes older APKs. Although datasets, such as AndroZoo and VirusShare, which are used to create our dataset, already use VirusTotal results for application classification, these provided results may change over time. (e.g. as a result of updates to the engine, with the primary objective of enhancing detection capabilities or due to the addition or removal of

engines from the platform).

The final dataset consists of 200K Android apps over a period of 10 consecutive years, from 2013 to 2022. For each year, 10k goodware and 10k malware apps are collected. To the best of our knowledge, we are leveraging the largest dataset of malware ever used in a longitudinal study on Android malware detection. For the sake of our study, it was decided to merge 7 years from 2013 to 2018 to form the training dataset, thus mimicking as best as possible the real world of the malware classifiers. Spanning 7 years ensures the use of 140k apps for training. Moreover, the apps from the 2019 to 2022 date range were intended solely to be used for evaluation during the longitudinal experiments.

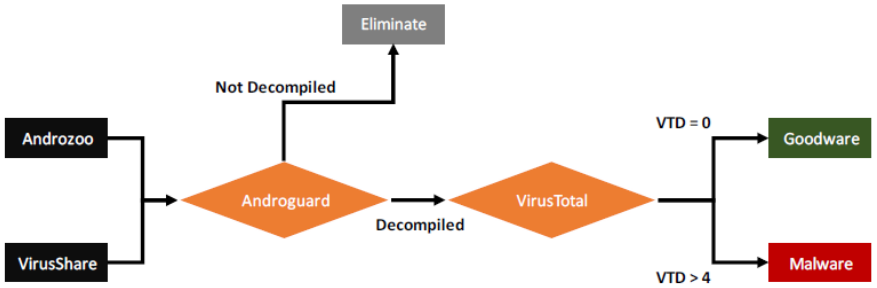


Figure 2. Dataset cleaning phases.

4.2 Feature Extraction

In this section, the entire feature-extraction process is described in detail, from the de-compilation of the APK files to the extraction of the different graphs and the construction of the adjacency matrices.

4.2.1 API Usage and Abstraction

API calls play an essential role in the operation of Android applications. They enable developers to access operating system functionalities and interact with hardware and software components. In our study, we distinguish between two types of APIs: built-in APIs (Android system APIs) and user-defined APIs (APIs defined by application developers). Exclusively in this work, we consider only the built-in APIs, as they represent consistent patterns and are available across different Android versions and devices, making them reliable indicators of app behavior. Motivated by the enhancement of the robustness of LongCGDroid, these API calls are abstracted to three different modes: package, class and method levels to cope with the changes of the Android system. The intuition behind this is that abstraction provides resilience to API changes in the Android framework, as classes and packages are added and removed less frequently than single API calls at the method level. Therefore, focusing only on specific methods can make the data less comparable and consistent over an extended period of time. For each API call extracted from the scanned app, it is processed into three abstractions, as illustrated in Figure 3. An API such as "android.telephony.SmsManager.sendTextMessage()" at the method level is abstracted to its class level as "Android.telephony.SmsManager", where the class part is preserved and to its package level as "Android.telephony", where the package part is preserved. These abstractions allow us to group together similar APIs and simplify data representation.

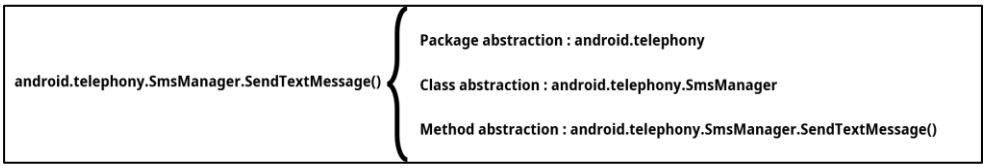


Figure 3. API abstraction modes.

4.2.2 Graph Extraction

Examining the graph call API can help effectively reveal the intention of an app. That depicts the calling relationships between different functions within a program. It shows how functions call other functions, creating a hierarchical structure of function calls. There are several tools for generating call graph APIs, such as Androguard or Flowdroid. Unfortunately, the call graph generated by these existing tools lacks inter-procedural built-in API calls. This is why LongCGDroid uses a custom

version of Androguard to extract the call graph API and generate the adjacency matrix.

To better clarify the differences between what is done by the original Androguard and our custom version, we employ a running example using a real-world malware sample. Specifically, Figure 4 lists a class extracted from the decompiled APK of malware disguised as a "*camera photo taking and editing*" (with package name *com.cp.camera*), which contains a remote app-related string. It is used to intercept SMS content for premium SMS fraud. To ease presentation, we focus on the portion of the code snippet executed in one function "loginByPost".

```
public String loginByPost(String code) {
    String str = Build.VERSION.RELEASE;
    String str2 = Build.MODEL;
    String phoneNumber = getPhoneNumber();
    String deviceId = getDeviceId();
    try {
        HttpURLConnection urlConnection =
            (HttpURLConnection) new URL(
                "http://139.59.107.168:8088/appsharejson?code=" + code)
                .openConnection();
        urlConnection.setRequestMethod("POST");
        urlConnection.setReadTimeout(5000);
        urlConnection.setConnectTimeout(5000);
        urlConnection.setDoOutput(true);
        urlConnection.setDoInput(true);
        if (urlConnection.getResponseCode() != 200) {
            return "error";
        }
        InputStream is = urlConnection.getInputStream();
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        byte[] buffer = new byte[1024];
        while (true) {
            int len = is.read(buffer);
            if (len != -1) {
                baos.write(buffer, 0, len);
            } else {
                is.close();
                baos.close();
                return new String(baos.toByteArray());
            }
        }
    } catch (Exception e) {
        e.printStackTrace();
        return "error";
    }
}
```

Figure 4. Malicious code (*com.cp.camera*) intercepting SMS content for premium SMS fraud.

First, the function gathers device-specific information, including the operator's name and phone number, through the *TelephoneManager* object, which gives information about the telephony services on the phone. Then, this app uploads all gathered information to the server (i.e., http://139.59.107.168:8088/). This malicious app receives the content of the message to send from the user's phone as a Json file.

The resulting sub-call graph generated by the original Androguard of the *loginByPost* function is shown in Figure 5(a). On the one hand, all the green nodes represent the user-defined APIs and the red nodes represent the built-in APIs called by the current function. On the other hand, the green arrows represent all links between user-defined APIs, the caller-callee relation, but the red nodes are not connected. No relationship is reported between built-in APIs by Androguard. This is why we use our custom Androguard version to represent the links between built-in APIs, as we can see in Figure 5(b), where red arrows are added in the sub-call graph during the pseudo-dynamic analysis to represent the caller-callee relations between built-in APIs. From this, we can infer that the

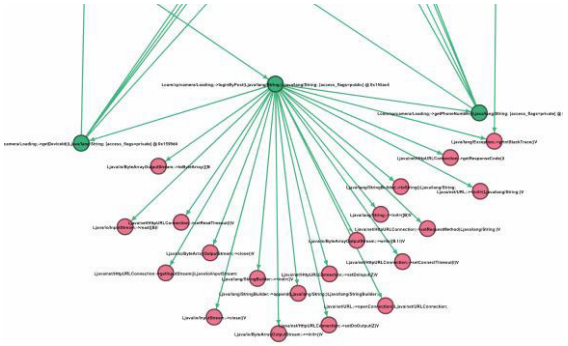


Figure 5 (a). API call graph extracted with custom original Androguard.

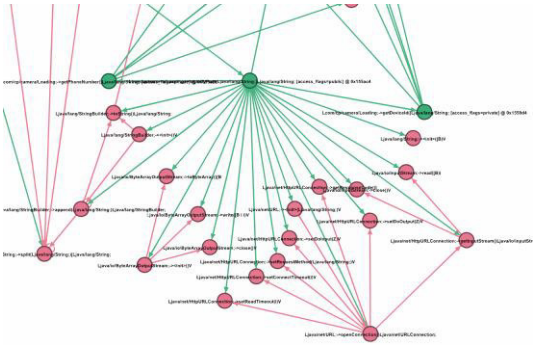


Figure 5 (b). API call graph extracted with Androguard highlighting external links.

loginByPost function calls the external method *java.net.url.<init>*, then the external method

java.net.url.openconnection is called and so on. These links are used to construct the adjacency matrices. This is made possible by using CFGs and DFGs to enhance the semantics of the call graph.

4.2.3 Pseudo-dynamic Analysis

We first construct a CFG for each method defined in an APK by using Androguard. We have customized the CFG extraction in order to return the graph with a Graph Modeling Language (GML) representation, which is more maintainable than the default proposed formats (i.e., png, jpg, raw). A CFG is defined as a directed graph G with the quadruple (N, E, S, F), where N is a finite set of nodes, $E \subseteq N \times N$ is a finite set of edges, where (n, m) ∈ E, if m may execute directly after n, $S \subseteq N$ is the set of starting nodes and $F \subseteq N$ is the set of exiting nodes.

We construct a CFG for each method in the call graph (Figure 6). This CFG will be used as input for the DFG. We focus on logging and tracking the built-in API calls, as they represent the features used to construct the adjacency matrix for each APK. Rather than executing the code decompiled from the c*lasses.dex* file in an emulator or sandbox, our analyzer tracks the code instruction by instruction without execution. During parsing, the analyzer uses the smali code to follow the DFG of the program instructions and register the update in a manner that mirrors a possible execution of a program, but does not compute any state information for the program.

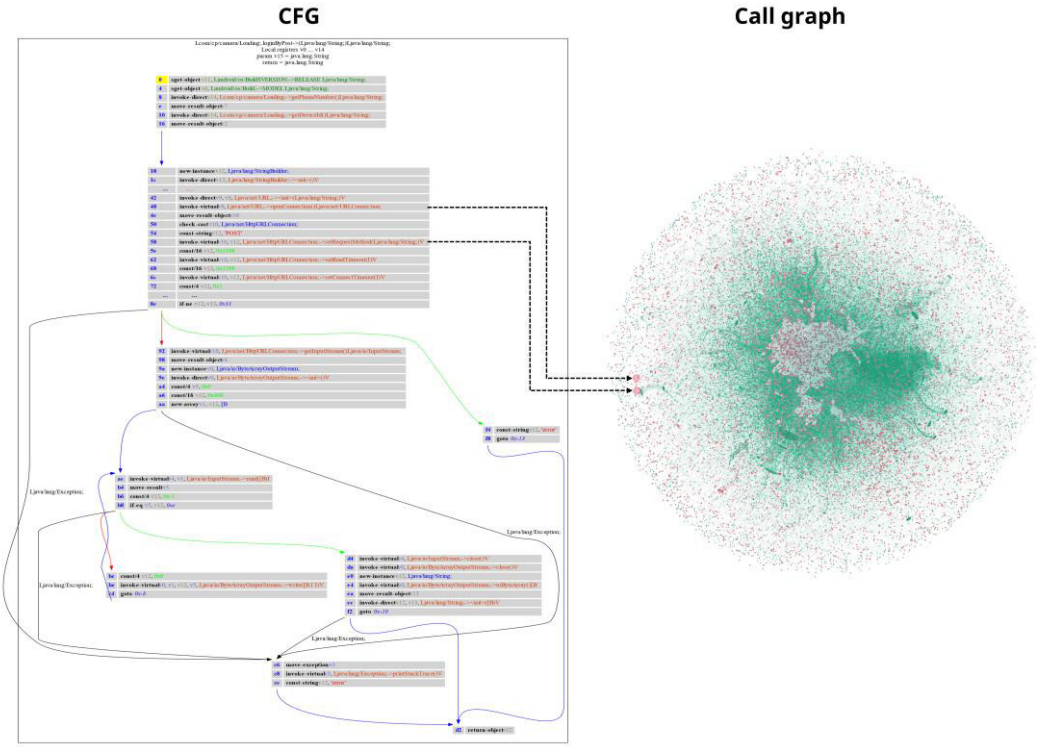


Figure 6. The CFG with smali code of the method loginByPost.

By generating the program-flow analysis through static analysis, we obtain a complete code coverage and avoid the possibility of dynamic malware obfuscation. To construct the DFG, our analyzer tracks three main categories of instructions: Regular instructions, register instructions and invoke instructions. The algorithm in Figure 7 gives the details of the extraction. This algorithm takes the CFG of a defined function *f* as input, represented by all possible paths and then returns the DFG of this function is represented as caller-callee relations.

Regular instructions include all instructions that do not affect the registers (e.g. *checkcast*, *instance-of*, …*etc.*); the analyzer simply steps over them with no state change, except for the update of the program counter for the length of the instruction. Register instructions are the instructions that store and manipulate data on registers during Android app execution (e.g. *move-result-object, move-object, ...etc.*). The registers' state is initialized by the methods' parameters and updated throughout the possible execution path. Invoke instructions are instructions used during an

invocation on the smali code (e.g. *invoke-virtual*, *invoke-virtual/range*, … *etc.).* They are used by LongCGDroid to extract the relation between built-in API calls as the edge of the DFG; that is, if an instruction I uses a register R, the value of which is already inferred by the register instruction as instruction J, then there is an edge from I to J.

For the sake of comprehension, we can take as an example the smali code of the function *loginByPost* in Figure 6. The instruction at the offset 0x58 is an invoke-virtual, which takes as arguments two registers, v10 and v12, and calls the function *setRequestMethod()*, as the register v10 is already inferred earlier during the analysis from the instruction 0x4e *move-result-object* and stated as the result of the instruction before, which is an invoke-virtual of the method *openConnection()*. From this, we can state that the *openConnection()* method as a caller calls the method *setRequestMethod()* as a callee. One can see this relation in the extracted call graph (Figure 6), marked by arrows with broken lines.

```
ALGORITHM: EXTRACT EXTERNAL API LINKS
    Input: CFG of the used function
    Output: Construction of the list of Caller-callee relation
1   list_callers_callees = ø
2   registers_state = initialize(method_description)
3   for each path in possible_paths
4       for each instruction in instructions_path
5           if instruction in register_instructions
6               registers_state = update_registers(instruction)
7           end if
8           if instruction in invoke_instructions
9               inferred_caller = get_caller(registers_state, register_number)
10              callee = get_callee(instruction)
11              list_callers_callees(inferred_caller, callee)
12          end if
13      end for
14  end for
```

Figure 7. Algorithm to extract built-in API links.

4.3 Graph Encoding

Once the caller-callee relations are inferred from an APK, it becomes straightforward to represent a graph as a matrix, such as the adjacency matrix. In more detail, given the list of caller-callee relations, we encode it into matrix *A* as follows: for each relation (n1; n2), we activate the element A[n1][n2] by 1, where *A* is initialized as a zero matrix. However, we have to consider three main factors:

1. **Graph Node Consistency**: This entails maintaining similarity in node values across different graphs, while also preserving equal node counts between these graphs.
2. **Matrix Size**: To ensure the uniformity of matrix sizes, the extracted nodes from each graph must exhibit consistency. Optimal matrix sizes, implying a relatively small number of nodes, are essential to facilitate swift processing and eliminate sparse matrix concerns.
3. **Feature Balance:** Balancing high-variance attributes across APK samples while avoiding matrices with inadequately activated regions is crucial. Consequently, emphasis is placed on built-in APIs to ensure consistent matrix sizes, while excluding user-defined methods.

According to the Android platform [49], there are 1081 packages, 4,853 classes and 43,800 methods in the Android API level 32. One cannot use all of these defined APIs to construct an adjacency matrix. If we take the case of packages only, the association rules that will be generated are 1167480 associations. In order to speed up the creation of adjacency matrices and avoid noisy structures, we selected different numbers of features for the different abstraction modes. For the package, class and method abstractions, we take 100, 200 and 300, respectively, from the most commonly used built-in

APIs to form a matrix. The original graphs would be trimmed by the selected features, resulting in a simplified version of the adjacency matrices with the chosen sizes. This is an empirical process, as we tried different feature lengths. When the number of features for method abstraction is small (e.g., 50×50) there was a sign of learning degradation, while when the number of features is high (e.g., over 1000 × 1000), the extraction phase would take much longer. The scheme of the process was the same for all different abstraction modes. This allows a trade-off between speed and learning performance that significantly improves the recognition rate. This selection was based on a statistical study designed to identify the most representative APIs used by malware and goodware apps. Figure 8 shows the top 10-API occurrences with the highest difference in usage between malware and goodware apps.

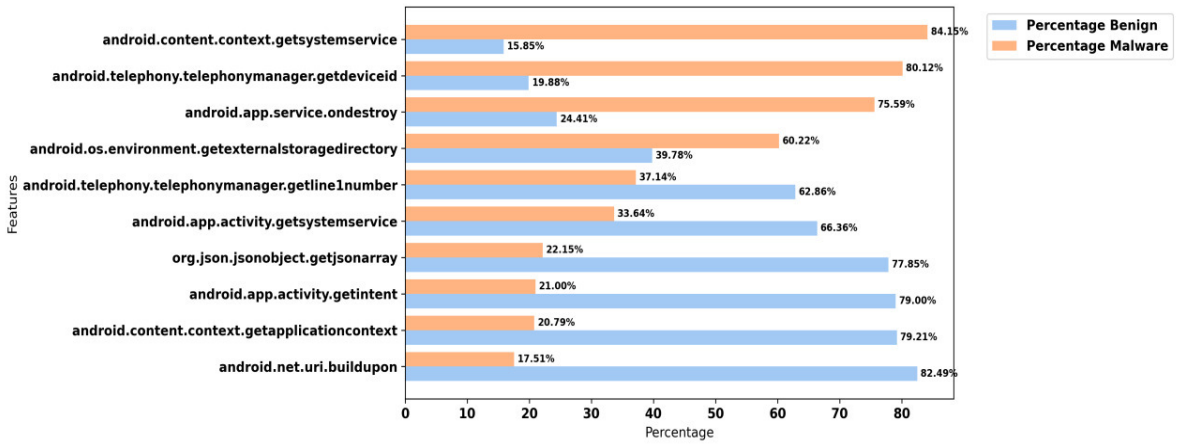


Figure 8. Frequency of the top 10 API with the highest difference between malicious and benign apps.

Once the features have been selected, the graph is converted into a 2-D adjacency matrix: the nodes represent rows and columns and if there is a connection between two nodes, the corresponding position in the matrix will be activated. As a result, each graph would form a matrix with several activated regions. Finally, we generate black-and-white images. Figure 9 presents the three different image representations of benign and malicious Android apps for each of the three abstraction modes.

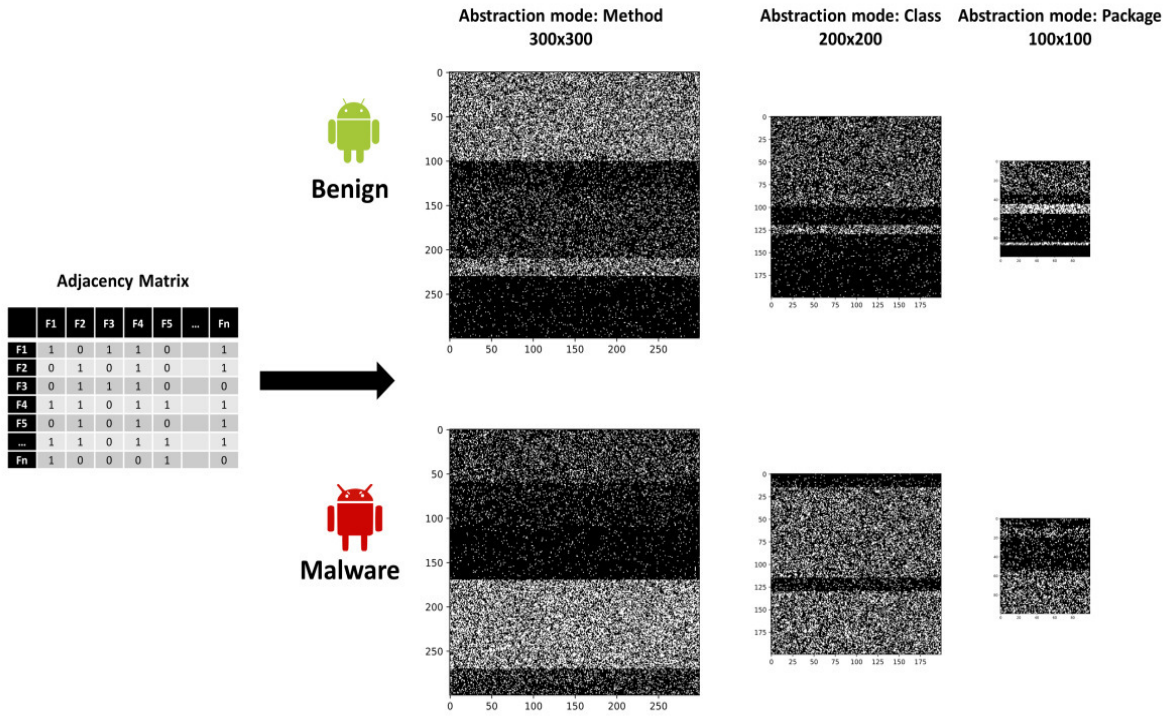


Figure 9. 2-D black and white images of benign and malicious apps in the three different abstractions.

4.4 Model Training

In this study, we compare and evaluate the longitudinal performance of machine-learning and deep-learning classifiers. To evaluate the features obtained by LongCGDroid, we applied them to the following classifiers: Machine-learning classifiers: DT, KNN, LR, RF and SVM; deep-learning classifiers: CNN and RNN-LSTM. For comparison, we also consider the API abstraction modes: method abstraction, class abstraction and package abstraction. For model construction and evaluation, the dataset is divided into training and test partitions. In order to obtain unbiased results, the test partition is always kept as a completely separate set and is never used for training or for feature engineering processes (extraction or pre-processing). The model parameters are selected using standard k-fold cross-validation (with k = 5) within the training set and following a grid search approach.

4.5 Evaluation Metrics

For this work, we considered a number of evaluation metrics that are commonly used in ML Android malware detection [3]. Note that some of these metrics, such as True Positive Rate (TPR), which is the same as Recall and False Positive Rate (FPR) or Precision, provide complementary information and should be used together to fully understand a system's performance. TPR indicates the rate of correct classification of malicious applications, while FPR indicates the rate of misclassification of clean (benign) applications. These metrics rely on: True Positives (TP), indicating the number of correct positive predictions and False Positives (FP), considering the number of incorrectly predicted negative items. We also used metrics such as accuracy and F1-score, Table 1.

Table 1. Evaluation metrics.

Metrics	Description
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	$\frac{TP}{TP + FP}$
TPR(Recall)	$\frac{TP}{TP + FN}$
F1-score	$\frac{2 \times Precision + Recall}{Precision + Recall}$

5. EXPERIMENTS AND RESULTS

We test the performance of the different models using behavioral-relationship features. Three series of tests are conducted in order to confirm the hypothesis that it is not enough to test a model with traditional methods, but it must also be evaluated longitudinally in order to ensure its robustness. In the first set of experiments, we analyze the accuracy of LongCGDroid by ignoring the evolution of the apps. In the second set of experiments, we conduct a longitudinal comparison of different models to evaluate their robustness. In the third set of experiments, we compare under the same conditions the LongCGDroid against eight state-of-the-art baseline detectors that rely on API calls.

During this comparison, we take into consideration the three different abstraction modes, denoted as P for package abstraction, C for class abstraction and M for method abstraction.

5.1 Experiment 1: Base Line Training Using a Temporally Imbalanced Dataset

Before analyzing the scenario related to longitudinal analysis, we test the classifiers under the basic assumption, using the whole dataset: 70% for training and 30% for testing. We have to notice that the dataset will be imbalanced with respect to time (years). Despite the fact that we take 70% of all malware and 70% of all goodware, keeping the remaining 30% of the samples for testing purposes, the number of apps per year is not controlled, which can create an imbalanced effect. The aim of this scenario is to mimic the conditions that are commonly assumed in the literature.

Table 2 provides an overview of the detection results achieved by LongCGDroid with

different classifiers. For comparison, we also consider the three abstraction modes: P, C and M.

Table 2. Performance evaluation for models trained in different modes with the whole dataset.

Classifier	Abstraction	Accuracy (%)	Precision (%)	TPR (%)	FPR (%)	F1-score (%)
CNN	***P***	92.93	93.6	92.17	6.3	92.88
	C	96.01	95.63	96.43	4.4	96.03
	M	99.42	99.32	99.52	0.68	99.42
SVM	***P***	92.68	92.52	92.86	7.5	92.69
	C	94.98	95.01	94.95	4.98	94.98
	M	97.76	97.24	98.32	2.79	97.77
RF	***P***	93.09	93.07	93.12	6.93	93.09
	C	94.69	94.68	94.71	5.31	94.69
	M	97.99	97.45	98.56	2.57	98
LR	***P***	92.74	92.56	92.96	7.47	92.76
	C	91.41	91.23	91.63	8.8	91.43
	M	93.04	92.77	93.36	7.27	93.06
RNN	***P***	90.32	90.29	90.36	9.71	90.32
	C	90.26	90.25	90.28	9.75	90.26
	M	92.77	92.49	93.1	7.55	92.79
KNN	***P***	90.99	90.88	91.13	9.14	91
	C	90.16	90.11	90.23	9.9	90.17
	M	90.37	90.16	90.62	9.88	90.39

The analysis of the presented classifiers and their performance across various abstraction levels yields valuable insights into their effectiveness for Android malware detection. First, as we can see in Figure 10, the abstraction modes P, C and M are all effective for malware detection. Whatever the classifier, all the metrics: Accuracy (Figure 10(a)), Precision (Figure 10(b)), TPR (Figure 10(c)) and F1-score (Figure 10(d)) exceed 90%. Among the tested classifiers, CNN emerges as a robust contender, demonstrating consistently strong results across all three levels of feature abstraction. In particular, we note the CNN's ability to maintain both higher TPRs (Figures 10(c)) and F1-scores (Figures 10(d)), accentuating its accuracy in reducing false positives and maximizing true positives, while maintaining a strong balance between precision and TPR.

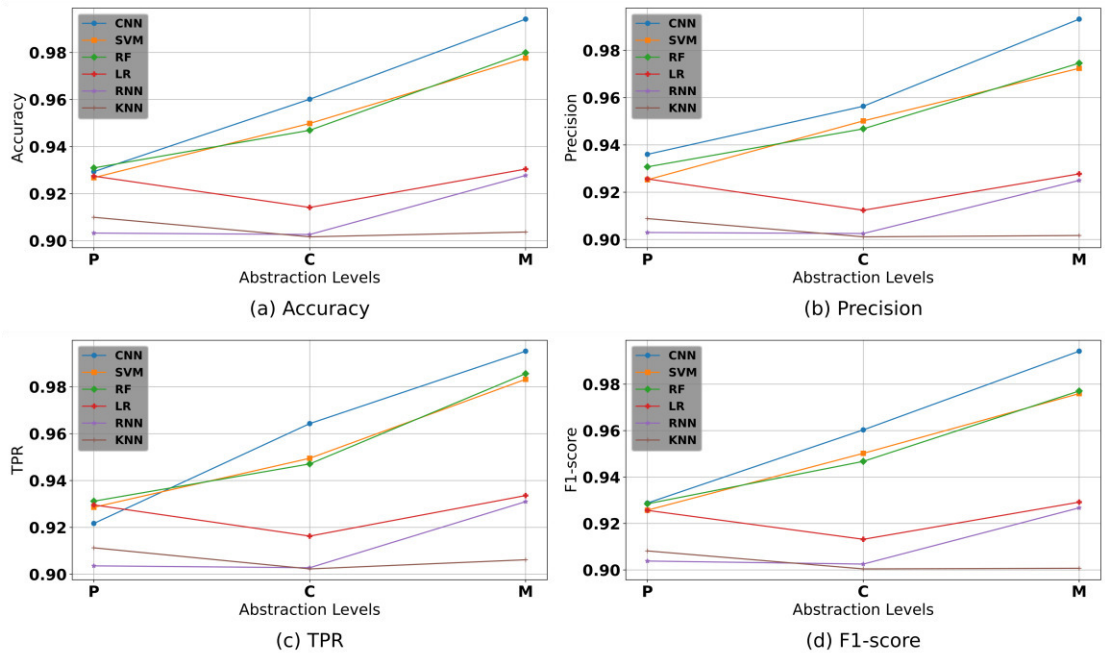


Figure 10. Results of the comparison of LongCGDroid with all classifiers and all abstractions.

This is most noticeable in the M abstraction, where the CNN has an F1-score of 99.42%. On the other

hand, the SVM and RF classifiers display performances close to those of the CNN. The SVM, for example, maintains consistent precision and TPR measurements, which indicates its reliability at different levels of abstraction. The RF classifier proves to be more sensitive to changes in abstraction, its F1-score varying from 93.09% with the P abstraction to 97.99% with the M abstraction, which indicates a dependence with regard to the level of abstraction chosen for the characteristics. The LR classifier shows less competitive results, with its F1-score almost constantly around 93%, which suggests its inability to find a balance between accuracy and TPR. Lastly, the RNN and KNN classifiers show the lowest results, whatever the level of abstraction used. Thus, CNN, SVM and RF all appear to be promising candidates, especially when high TPR and F1-score are important. However, no conclusions can be drawn about the robustness of the models.

5.2 Experiment 2: Longitudinal Performance of LongCGDroid

In this longitudinal study, all the classifiers are trained on the dataset from 2013 to 2018 and evaluated on the remaining part, from 2019 to 2022. The performance of various machine-learning and deep- learning classifiers was assessed across different levels of abstraction, as was the case with experiment 1.

Starting with the results from the first year of test 2019, it can be observed from Figure 11 that across all levels of abstraction, the CNN consistently achieved high TPR, Precision and F1-score values. Particularly, when considering the C abstraction, the CNN demonstrated remarkable performance with TPR and F1-score around 97%. The SVM showcased competitive results, although its performance slightly degraded with the increased abstraction. RF exhibited stable performance across different abstraction levels, while RNN and KNN demonstrated decreased results, with RNN showing relatively better performance in precision than KNN.

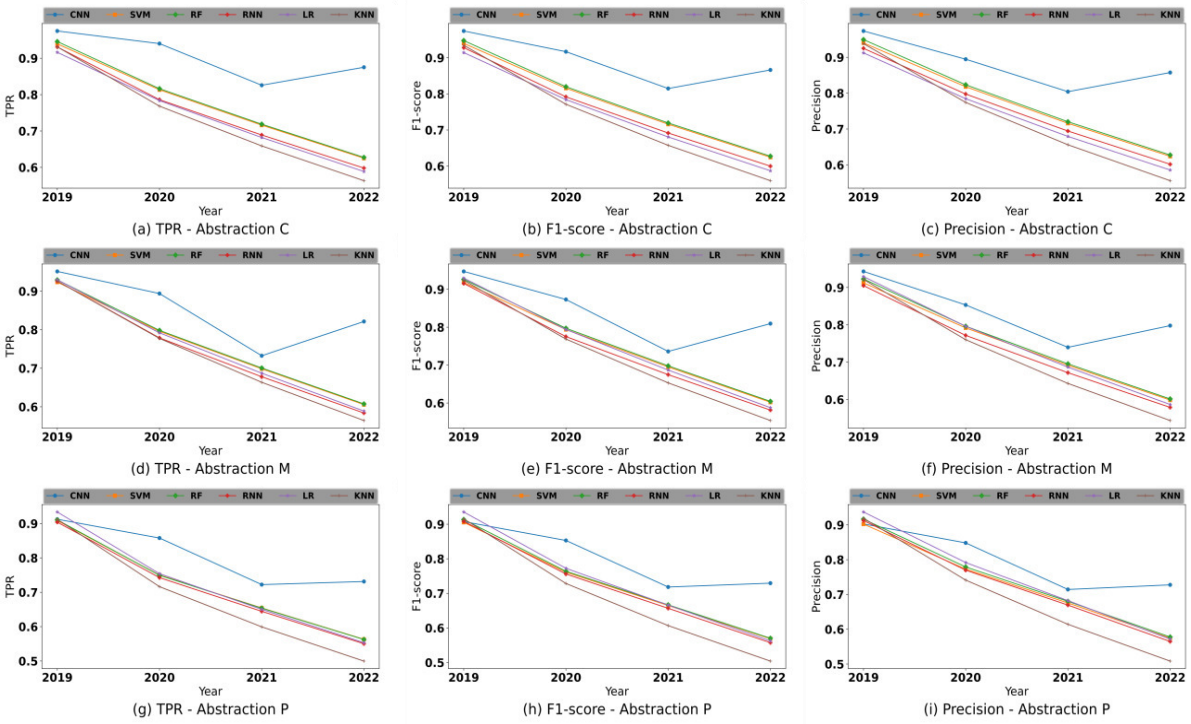


Figure 11. Results of the longitudinal comparison of LongCGDroid with all classifiers.

In the subsequent years, the trends shifted. By 2020, there is a general decrease in performance for most classifiers, especially in terms of accuracy and precision. However, CNN continued to outperform other classifiers in terms of TPR and F1-score, even though its precision has dropped. SVM, RF and RNN showcased consistent trends across the years, with SVM's performance being stable, but comparatively lower, RF maintaining relatively strong performance and RNN's accuracy and precision experiencing a slight decrease. This degradation is more important in package and method abstraction than in class abstraction.

Moving to 2021, there was a continued decline in overall performance. Notably, the decline was

in the accuracy and precision of all classifiers. The performance gaps between classifiers became narrower and SVM's performance even improved in terms of precision with class and method abstractions. This year marked a more competitive landscape among the classifiers.

Finally, in 2022, the performance of CNN evolved once again. However, it shows signs of decline with P and M abstractions. SVM and RF exhibited a consistent trend with slight performance dips, while the performance of RNN and KNN showed more fluctuations.

The results show a consistent decline in model performance at all levels of abstraction. However, a notable observation is the gradual decline observed over time when using class abstraction. This particular abstraction shows a trend toward better results during the four-year post-training period. Class abstraction appears to be more resilient to API changes than the highest level of abstraction; namely, method abstraction. In contrast, package abstraction continues to show promising model performance. However, it is clear that these models lose stability over the years. Therefore, their effectiveness in representing the dataset is somewhat compromised.

5.3 Experiment 3: LongCGDroid *vs.* State-of-the-art Static Baseline Systems

To demonstrate the effectiveness of our proposed method in addressing malware evolution, we conducted a comparative study with eight state-of-the-art baseline systems; namely, Maldozer, DroidSieve, RevealDroid, MamaDroid, DroidEvolver, Admat, which are static approaches' classifiers, and DroidSpan and DroidCat, which are dynamic approaches' classifiers. We specifically choose to compare with approaches that (1) employ API calls to perform detection, (2) are fully designed considering the evolution of the apps and (3) are longitudinally evaluated. These criteria apply to Maldozer, DroidSieve, RevealDroid, MamaDroid, DroidEvolver, Admat, DroidSpan and DroidCat, except for DroidSieve and AdMat, which are not longitudinally evaluated in the original works. For consistency with the evaluated systems, we consider the configurations that produced optimal results in the respective original studies. For instance, it's important to highlight that MaMaDroid employs two distinct abstraction modes: family and package. The family mode operates at a more refined granularity level compared to the package mode, resulting in a smaller count of API calls, specifically nine families. On the other hand, Maldozer adopts only the abstraction method mode. Specifically, this involves employing the family abstraction combined with the RF model for MaMaDroid and utilizing the method abstraction for Maldozer. Regarding our approach, we will take into account the CNN model with the class abstraction. During this experiment, a subset of the dataset from 2013 to 2018 is used for feature extraction and training, while the remainder from 2019 to 2022 is used for evaluation. We focused our analysis on key performance indicators, specifically TPR and the F1-score.

The results indicate a notable consistency in the performance of LongCGDroid, especially when compared to other systems. One year after training, LongCGDroid outperformed all other systems with a TPR of 97.49% (Figure 12(a)) and a relatively low FPR of 2.66% (Figure 12(b)), achieving an F1-score of 97.42% (Figure 12(c)). MaMaDroid, DroidEvolver, DroidSpan and DroidCat also showcased commendable performance with F1-scores of 95.08%, 94.40%, 92.46% and 91.30%, respectively, which quantify their ability to correctly identify malware instances, while DroidSieve and RevealDroid lagged with an F1-score of 60.31% and 72.98%, respectively. AdMat, on the other hand, had a performance with an F1-score of 88.08%, placing it in the mid-range among the compared systems.

As we progress over the years, we observe a gradual performance decrease, as shown by the changes in the F1-score, with LongCGDroid still maintaining a leading position with an F1-score of 91.69% in 2020. Maldozer, DroidEvolver, MaMaDroid, DroidSpan and DroidCat followed closely with F1-scores of 88.57%, 87.56%, 86.94%, 82.73% and 85.67%, respectively. The performance gap between LongCGDroid and other systems like DroidSieve, RevealDroid and Admat widened, as they reached F1-scores of 41.67%, 66.49% and 77.54%, respectively. The trend of declining performance continued into 2021, with LongCGDroid recording an F1-score of 81.48% and all the other systems experiencing a decline to values below 80%. They appear to be significantly more impacted by the reduction in TPR. However, the performance variations among the systems were more apparent, with LongCGDroid maintaining its competitive edge. These results accentuate the resilience of the LongCGDroid. Figure 12(b) does not follow clear, decreasing trends,

as we can notice a little increase in the year 2022 on all the systems; this can be explained by the fact that at some point, an old behavior becomes popular again, leading to a sudden increase in the performance of all detectors.

We can see that DroidEvolver, MaMaDroid, DroidSpan and DroidCat remain more longitudinally competitive in comparison with our LongCGDroid system than the other systems. This can be attributed to the fact that MaMaDroid uses a lower granularity API family, DroidEvolver continually updates its feature set as an online learning system and both DroidSpan and DroidCat employ dynamic features to profile the apps, hence enhancing the characterization of their behavior, unlike Maldozer, which relies on a method-level abstraction. On the other hand, DroidSieve and AdMat, which were not initially designed to consider the evolution of the apps, displayed a consistent downward trend in their F1-scores, indicating diminishing effectiveness over the years. Although DroidSieve's and RevealDroid's initial study suggests that they are very resistant to obfuscation, their characteristics do not necessarily make them suitable choices for overcoming the evolution of apps. The decline and instability shown in the AdMat can be attributed to its exclusive reliance on the CFG inferred from the APIs without incorporating the DFG derived from pseudo-dynamic analysis, as is the case with LongCGDroid.

LongCGDroid is a static-based system that uses pseudo-dynamic analysis to extract Android API features, which provides an efficient mechanism to abstract significant characteristics of the Android applications under study, thus allowing us to extract Android API features efficiently while minimizing the overhead typically associated with real dynamic-analysis methods. However, using a real dynamic analysis could provide a more accurate reflection of the run-time behavior of the applications under analysis. The pseudo-dynamic analysis may not capture the real-time interactions and the exact execution flow in a similar manner as real dynamic analysis would. For instance Android malware may disguise its malicious behavior by only triggering a malicious API call after receiving a specific network signal. Under the current pseudo-dynamic analysis, such behavior might go unnoticed, as the analysis relies on a predetermined application state. However, with dynamic analysis, the system can observe the application's behavior in a controlled, yet realistic, execution environment, recording how it interacts with system and user data through API calls when stimulated by external triggers.

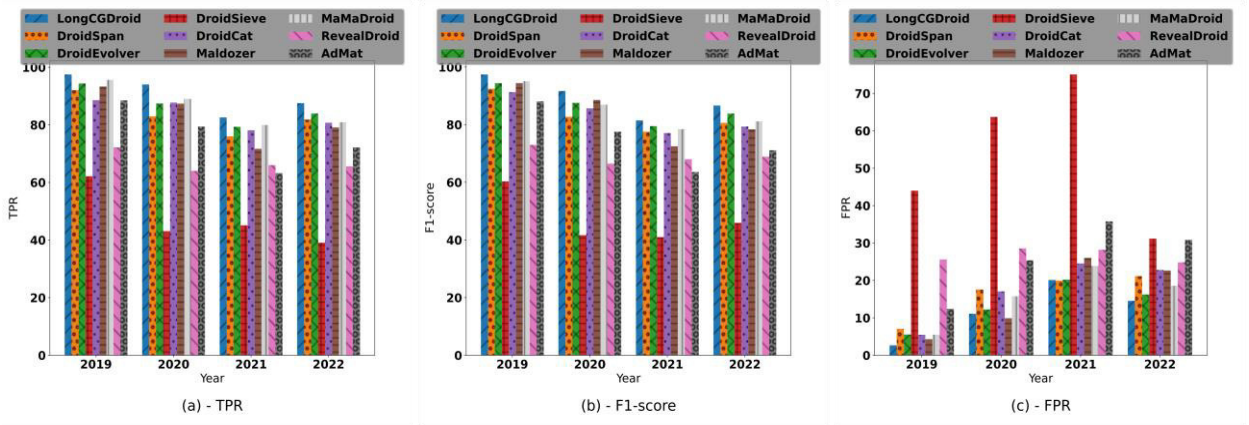


Figure 12. Performance evolution of the systems across the years.

6. CONCLUSION

In this work, we conducted a comparative longitudinal study between machine-learning and deep-learning algorithms using our adjacency matrix-based tool, LongCGDroid. Through our experiments using different levels of abstraction, we observed that the usual evaluation lacks information about model robustness. However, employing a longitudinal approach allows us to infer various details regarding model robustness and evolution. Thus, we observed that while the method abstraction yields better results in the usual approach, for maintaining higher robustness, it is advisable to focus on class abstraction. Class abstraction, being more resilient, enables balanced detection performance. This resilience has proven to be beneficial for the deep-learning CNN classifier for the sake of conserving a good balance of detection over time. Our method is also

compared with eight state-of-the-art baseline research works; our approach has proven to be more resilient over the years.

Future work may investigate the use of RGB color images to get more inferred details. This can have an impact on mitigating the diminishing performance of the classifiers of machine learning and deep learning-based malware detectors.

ACKNOWLEDGEMENTS

The authors would like to thank Salim Grabssi, the Director of the Computing Center of the University of Boumerdes, for letting them conduct some of the computing for this project.

REFERENCES

[1] Statista.com, "Mobile Operating Systems' Market Share Worldwide from January 2022 to January 2023," [Online], Available: https://www.statista.com/statistics/272698/global-market-share-held-by-mobile- operating-systems-since-2009/.

[2] Security List by Kaspersky, "IT Threat Evolution Q1 2023, Mobile Statistics," [Online], Aailable: https://securelist.com/it-threat-evolution-q1-2023-mobile-statistics/109893/.

[3] W. Wang et al., "Constructing Features for Detecting Android Malicious Applications: Issues, Taxonomy and Directions," IEEE Access, vol. 7, pp. 67602-67631, 2019.

[4] Y. C. Shyong, T. H. Jeng and Y. M. Chen, "Combining Static Permissions and Dynamic Packet Analysis to Improve Android Malware Detection," Proc. of the 2nd Int. Conf. on Computer Communication and the Internet (ICCCI), pp. 75-81, Nagoya, Japan, 2020.

[5] L. Li, T. F. Bissyandé, M. Papadakis et al., "Static Analysis of Android Apps: A Systematic Literature Review," Information and Software Technology, vol. 88, pp. 67-95, 2017.

[6] H. Cai, "Embracing Mobile App Evolution *via* Continuous Ecosystem Mining and Characterization," Proc. of the IEEE/ACM 7th Int. Conf. on Mobile Software Engineering and Systems, pp. 31–35, Seoul, Korea, 2020.

[7] H. Cai and B. Ryder, "A Longitudinal Study of Application Structure and Behaviors in Android," IEEE Transactions on Software Engineering, vol. 47, no. 12, pp. 2934-2955, 2021.

[8] H. Cai, X. Fu and A. Hamou-Lhadj, "A Study of Run-time Behavioral Evolution of Benign *versus* Malicious Apps in Android," Information and Software Technology, vol. 122, p. 106291, 2020.

[9] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder and L. Cavallaro, "TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time," Proc. of the 28th USENIX Conf. on Security Symposium, pp. 729–746, 2019.

[10] L. Nguyen-Vu, J. Ahn and S. Jung, "Android Fragmentation in Malware Detection," Computers & Security, vol. 87, p. 101573, 2019.

[11] G. Suarez-Tangil and G. Stringhini, "Eight Years of Rider Measurement in the Android Malware Ecosystem: Evolution and Lessons Learned," IEEE Transactions on Dependable and Secure Computing, vol. 19, pp. 107-118, 2018.

[12] A. Guerra-Manzanares and H. Bahsi, "On the Relativity of Time: Implications and Challenges of Data Drift on Long-term Effective Android Malware Detection," Computers & Security, vol. 122, p. 102835, 2022.

[13] F. Ceschin et al., "Fast & Furious: On the Modeling of Malware Detection As an Evolving Data Stream," Expert Systems with Applications, vol. 212, p. 118590, 2023.

[14] L. Onwuzurike et al., "MAMADROID: Detecting Android Malware by Building Markov Chains of Behavioral Models," ACM Transactions on Privacy and Security, vol. 22, no. 2, pp. 1-34, 2019.

[15] E. B. Karbab, M. Debbabi, A. Derhab and D. Mouheb, "Android Malware Detection Using Deep Learning API Method Sequences," Digital Investigation, vol. 24, pp. S48-S59, 2017.

[16] K. Xu et al., "DroidEvolver: Self-Evolving Android Malware Detection System," Proc. of the 2019 IEEE European Symp. on Security and Privacy (EuroS&P), pp. 47-62, Stockholm, Sweden, 2019.

[17] G. Suarez-Tangil et al., "DroidSieve: Fast and Accurate Classification of Obfuscated Android Malware," Proc. of the 7th ACM on Conference on Data and Application Security and Privacy, pp. 309–320, DOI: 10.1145/3029806.3029825, 2017.

[18] L. N. Vu and S. Jung, "AdMat: A CNN-on-Matrix Approach to Android Malware Detection and Classification," IEEE Access, vol. 9, pp. 39680-39694, 2021.

[19] J. Garcia, M. Hammad and S. Malek, "Lightweight, Obfuscation-resilient Detection and Family Identification of Android Malware," Proc. of the 40th Int. Conf. on Software Engineering, p. 497, DOI: 10.1145/3180155.3182551, 2018.

[20] H. Cai, N. Meng, B. Ryder and D. Yao, "DroidCat: Effective Android Malware Detection and Categorization *via* App-Level Profiling," IEEE Transactions on Information Forensics and Security, vol.

14, no. 6, pp. 1455-1470, 2019.
[21] H. Cai, "Assessing and Improving Malware Detection Sustainability through App Evolution Studies," ACM Trans. Softw. Eng. Methodol., vol. 29, no. 2, p. Article 8, 2020.
[22] N. Elenkov, Android Security Internals: An In-Depth Guide to Android's Security Architecture, ISBN-10: 9781593275815, No Starch Press, 2014.
[23] A. Desnos. "Androguard-reverse Engineering, Malware and Goodware Analysis of Android Applications," [Online], Aailable: https://github.com/androguard/androguard.
[24] S. Arzt et al., "FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps," SIGPLAN Not., vol. 49, no. 6, pp. 259–269, 2014.
[25] R. Nix and J. Zhang, "Classification of Android Apps and Malware Using Deep Neural Networks," Proc. of the Int. Joint Conf. on Neural Networks (IJCNN), pp. 1871-1878, Anchorage, USA, 2017.
[26] S. Y. Yerima and S. Khan, "Longitudinal Performance Analysis of Machine Learning-based Android Malware Detectors," Proc. of the 2019 Int. Conf. on Cyber Security and Protection of Digital Services (Cyber Security), pp. 1-8, Oxford, UK, 2019.
[27] J. Jung, H. Kim, D. Shin, M. Lee, H. Lee, S.-j. Cho and K. Suh, "Android Malware Detection Based on Useful API Calls and Machine Learning," Proc. of the 2018 IEEE 1st Int. Conf. on Artificial Intelligence and Knowledge Engineering (AIKE), pp. 175-178, Laguna Hills, USA, 2018.
[28] N. Peiravian and X. Zhu, "Machine Learning for Android Malware Detection Using Permission and API Calls," Proc. of the 2013 IEEE 25th Int. Conf. on Tools with Artificial Intelligence, pp. 300-305, Herndon, USA, 2013.
[29] M. Qiao, A. H. Sung and Q. Liu, "Merging Permission and API Features for Android Malware Detection," Proc. of the 2016 5th IIAI Int. Congress on Advanced Applied Informatics (IIAI-AAI), pp. 566-571, Kumamoto, Japan, 2016.
[30] A. H. E. Fiky, A. Elshenawy and M. A. Madkour, "Detection of Android Malware Using Machine Learning," Proc. of the IEEE Int. Mobile, Intelligent and Ubiquitous Computing Conf. (MIUCC), pp. 9- 16, Cairo, Egypt, 2021.
[31] M. Alazab et al., "Intelligent Mobile Malware Detection Using Permission Requests and API Calls," Future Generation Computer Systems, vol. 107, pp. 509-521, 2020.
[32] Z. Wang, K. Li, Y. Hu, A. Fukuda and W. Kong, "Multilevel Permission Extraction in Android Applications for Malware Detection," Proc. of the 2019 Int. Conf. on Computer, Information and Telecommunication Systems (CITS), pp. 1-5, Beijing, China, 2019.
[33] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an and H. Ye, "Significant Permission Identification for Machine Learning-based Android Malware Detection," IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3216-3225, 2018.
[34] C. Yang, Z. Xu, G. Gu, V. Yegneswaran and P. Porras, "DroidMiner: Automated Mining and Characterization of Fine-grained Malicious Behaviors in Android Applications," Proc. of the European Symposium on Research in Computer Security (ESORICS 2014), vol. 8712, pp. 163-182, 2014.
[35] T. E. Wei et al., "DroidExec: Root Exploit Malware Recognition against Wide Variability *via* Folding Redundant Function-relation Graph," Proc. of the 17th Int. Conf. on Advanced Communication Technology (ICACT), pp. 161-169, PyeongChang, Korea, 2015.
[36] A. Narayanan, L. Yang, L. Chen and L. Jinliang, "Adaptive and Scalable Android Malware Detection through Online Learning," Proc. of the Int. Joint Conf. on Neural Networks (IJCNN), pp. 2484-2491, Vancouver, Canada, 2016.
[37] Y. Wu, J. Shi, P. Wang, D. Zeng and C. Sun, "DeepCatra: Learning Flow- and Graph-based Behaviours for Android Malware Detection," IET Information Security, vol. 17, no. 1, pp. 118-130, 2023.
[38] T. Lei, Z. Qin, Z. Wang, Q. Li and D. Ye, "EveDroid: Event-aware Android Malware Detection against Model Degrading for IoT Devices," IEEE Internet of Things Journal, vol. 6, no. 4, pp. 6668-6680, 2019.
[39] J. McGiff et al., "Towards Multimodal Learning for Android Malware Detection," Proc. of the Int. Conf. on Computing, Networking and Communicat. (ICNC), pp. 432-436, Honolulu, USA, 2019.
[40] D. Li et al., "Opcode Sequence Analysis of Android Malware by a Convolutional Neural Network," Concurrency and Computation: Practice and Experience, vol. 32, no. 18, p. e5308, 2020.
[41] X. Sun et al., "Android Malware Detection Using Sequential Convolutional Neural Networks," Journal of Physics: Conference Series, vol. 1168, no. 6, p. 062010, 2019.
[42] N. McLaughlin et al., "Deep Android Malware Detection," Proc. of the Seventh ACM on Conf. on Data and Application Security and Privacy, pp. 301–308, DOI: 10.1145/3029806.3029823, 2017.
[43] T. H.-D. Huang and H.-Y. Kao, "R2-D2: ColoR-inspired Convolutional NeuRal Network (CNN)-based AndroiD Malware Detections," arXiv: 1705.04448 [cs.CR], 2018.
[44] P. Faruki, B. Buddhadev, B. Shah, A. Zemmari, V. Laxmi and M. S. Gaur, "DroidDivesDeep: Android Malware Classification *via* Low Level Monitorable Features with Deep Neural Networks," Proc. of the Int. Conf. on Security & Privacy (ISEA-ISAP 2019), vol. 939, pp. 125-139, 2019.
[45] K. Allix, T. F. Bissyandé, J. Klein and Y. L. Traon, "AndroZoo: Collecting Millions of Android Apps for the Research Community," Proc. of the 2016 IEEE/ACM 13th Working Conf. on Mining

Software Repositories (MSR), pp. 468-471, Austin, USA, 2016.
[46] VirusShare, "VirusShare.com - Because Sharing is Caring," [Online], Available: https://virusshare.com/.
[47] Virustotal, "Analyse Suspicious Files," [Online], Available: https://www.virustotal.com/.
[48] A. Salem, S. Banescu and A. Pretschner, "Don't Pick the Cherry: An Evaluation Methodology for Android Malware Detection Methods," arXiv: 1903.10560 [cs.CR], 2019.
[49] Android, "Android APIs Reference," [Online], Available: https://developer.android.com/reference/packages.

ملخص البحث:

تستهدف هذه الورقة البحثية مقارنة الأداء الطّولي بين مُصنِّفات تعلُّم الآلة ومُصنِّفات التّعلُّم العميق لكشْف البرمجيات الخبيثة في تطبيقات أَنْدرويد، باستخدام مستوياتٍ مختلفة من تجريد السِّمات. وباستخدام مجموعة بيانات لتطبيقات أَنْدرويد على مدى عشر سنوات (2013-2022).

نقترح طريقةً فعالةً قائمة على الصُّوَر لكشْف البرمجيات الخبيثة في تطبيقات أَنْدرويد. كذلك نعمل على تقييم الأداء الطّولي للطّريقة المقترحة، باستخدام نماذج تعلُّم الآلة ونماذج التّعلُّم العميق.

وقدّ بينت التّجارب تراجعاً متواصلاً في الأداء لجميع المصنِّفات عند تقييمها على عيّناتٍ من فتراتٍ متأخِّرة. واثبت نموذج التعلُّم العميق القائم على الشّبكات العصبية الالتفافية (CNN) تحت تجريد الصِّنْف قدْراً من الاستقرار مع الزّمن. وبمقارنة الطّريقة المقترحة بعددٍ من الطُّرُق المشابهة الواردة في أدبيات الموضوع، وأثبتت الطّريقة المقترحة تفوُّقها من حيث دِقّة الكشْف.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

FUSION OF DEEP LEARNING ARCHITECTURES FOR ENHANCED TARGET RECOGNITION ON SAR IMAGES

K. Cheikh[1], R. Aitahcene[1], A. Toumi[2] and Z. Hammoudi[1]

(Received: 26-Aug.-2023, Revised: 24-Oct.-2023, Accepted: 2-Nov.-2023)

ABSTRACT

In various applications of radar imagery, one of the fundamental problems is mainly linked to the analysis and interpretation of the images provided, in particular the recognition of moving and/or fixed targets. This task has become more difficult due to the large volume of radar data. This led to the use of automatic-processing and target-recognition methods. The aim of this study is to explore data fusion in SAR (Synthetic Aperture Radar) image classifiers. To this end, we propose a new approach to combine three CNN (Convolutional Neural Network) architectures with several fusion rules. First, we perform a training process of three deep-learning architectures; namely, the basic CNN, the Xception and the AlexNet architectures. Then, two fusion techniques are proposed. The first one deals with the majority rule and the second uses a neural network to combine the decision outputs obtained from three elementary classifiers to achieve the final decision. To evaluate and validate the proposed approach, the MSTAR (Moving and Stationary Target Acquisition and Recognition) dataset is used. The obtained performances of the fusion techniques improve the recognition rate with a final accuracy of 99.59% for the majority rule and 99.51% for the neural network-based rule, which surpasses the accuracy of each individual CNN.

KEYWORDS

Automatic target recognition, Synthetic aperture radar images, Deep learning, Decision fusion.

1. INTRODUCTION

One of the fundamental problems in radar imaging is mainly related to the analysis and interpretation of images acquired in various applications, including recognition of moving and fixed targets. This task becomes more difficult to achieve when automatic processing methods and decision-making are concerned in regard to the large volume of radar data and speckle measurements [1]-[6]. Regarding its global coverage as well as its weather independence, SAR (Synthetic Aperture Radar) imagery has great potentials for military and/or civilian surveillance. In order to leverage these potentials, ML (Machine Learning) can be used to automatically process large amounts of data for different goals, such as ATR (Automatic Target Recognition). For instance, target recognition in radar images presents an essential task for monitoring and surveillance of sensitive areas, such as military or/and civilian zones.

Several methods for target recognition and classification from radar images have been developed in the literature. Due to the absence of efficient feature presentation, interpretation and understanding of radar images, classification of radar images is a significant challenging task. An adequate feature-extraction approach, which can abstract spatial information from radar images and improve classification accuracy, is required. Feature extraction and target recognition in images have been of great interest for many years. Many methods have been proposed by many researchers [1], [7]. These include classical classification methods, such as Bayesian methods, SVM (Support Vector Machine) [8], AdaBoost (Adaptive Boosting) [9], decision trees, WSC (Weighted Sparse Classification), …etc. [10]-[11]. Although, these methods perform well in some situations, their performance degrades significantly in other situations and conditions. This has led researchers to adopt other more sophisticated approaches, such as ANNs (Artificial Neural Networks), which have a very efficient learning capability in a multitude of system-modeling problems [1]. In recent years, we have seen the emergence of DL (Deep Learning) methods with several variants. These latter have given very satisfactory results in several application areas, such as cybersecurity, text analysis, visual and image recognition [12]-[16] and many more. Several research studies have explored the use of RNNs (Recurrent Neural Networks) in the context of SAR image classification, with particular emphasis on

1. K. CHEIKH (Corresponding Autor), R. Aitahcene and Z. Hammoudi are with SISCOM Laboratory, Department of Electronics, University of Frères Mentouri, Constantine 1, Algeria. Email: khaireddine.cheikh@umc.edu.dz
2. A.Toumi is with Lab STICC UMR CNRS 6285, ENSTA Bretagne, Brest, France. Email: abdelmalek.toumi@ensta-bretagne.fr

LSTM (Long Short-Term Memory) networks. LSTMs are specifically designed to handle sequences of data or time series, a feature particularly relevant to the sequential nature inherent in SAR data [17]. To further enhance SAR classification performance, other researchers have delved into hybrid architectures that combine both CNNs (Convolutional Neural Networks) and RNNs (Recurrent Neural Networks). These hybrid approaches showed promising results in SAR image classification by enabling the simultaneous capture of spatial features and temporal dependencies [18]-[19]. With the constant evolution of deep-learning network architectures and their maturity, many researchers have turned to the use of pre-trained DNNs (Deep Neural Networks) in the context of SAR image classification. This approach explores how features pre-learned by these networks can be judiciously employed to significantly increase the accuracy of SAR image classification [20].

Other researchers have attempted to expand the scope of the training dataset by implementing various operations on the images. These operations include rotations, translations, changes in scale and the introduction of noise [21]-[23]. Furthermore, several super-resolution approaches based on DCNNs (Deep Convolutional Neural Networks) have been widely adopted to enhance image resolution. This increase in resolution has directly contributed to an improvement in classification accuracy [24]. Additionally, there are studies that have employed data-augmentation techniques utilizing GANs (Generative Adversarial Networks) to generate synthetic images [25]. Furthermore, in last years, advanced approaches have emerged, combining a DNN in order to improve and achieve more robust and accurate classification results. Furthermore, in recent years, advanced approaches have emerged, utilizing data fusion to significantly enhance SAR image classification. Among these methods, data fusion from multiple sensors has demonstrated a clear improvement compared to using data from a single sensor [26]. Another promising approach lies in the use of model ensembles for SAR image classification, which explores how the aggregation of multiple models can substantially increase classification accuracy. Commonly adopted methods include ensemble learning, where multiple classifiers are trained. Their predictions are combined using various techniques, including voting, weighting and stacking methods [27]-[28].

In this context, the main objective of this research work is to explore new combined architectures of DL to accomplish the classification task, using two methods. The first method relies on fusing the output data from each classifier to make a final decision using the majority fusion rule. The second approach involves the use of ANNs (Artificial Neural Networks) to build a data fusion model, thus exploiting the advantages of each DLN (DL Network) architecture used in this study. Both methods will be evaluated and validated on the MSTAR (Moving and Stationary Target Acquisition and Recognition) dataset [29].

The contribution of this work deals with the study of two fusion techniques. The first one is based on the majority rule and the second one relies on the NN-based rule, in order to improve the recognition performance.

The remainder of this paper is organized as follows. In Section 2, we introduce the CNNs and their variants. In section 3, we present the proposed method. In Section 4, we detail the training, describe and discuss some experimental results performed on the MSTAR dataset. In Section 5, we conclude our work and give some future perspectives.

2. CONVOLUTIONAL NEURAL NETWORKS

CNNs are to date the most efficient models for classifying images and particularly radar images. An input image is provided in the form of a matrix of pixels. It has a two-dimensional array for SAR image for each channel/layer. For a multi-layer image, such as color images (3 layers for Red, Green and Blue) or multi-spectral images, the input image is provided as a multi-dimensional arrays.

The first part of a CNN is the convolutive step, which operates as an image-feature extractor. The image is passed through a series of filters or convolution kernels, creating new images called convolution maps. Some intermediate filters reduce the size of the input data by a maximum pooling operation. Finally, the last convolution maps are laid flat and concatenated into a feature vector, called the CNN code. This CNN code is the input of a layer called the FC (Fully Connected) layer which is a multi-layer perceptron. Its role is the combination of the characteristics of the CNN code to classify the image. The output is the last layer, called SoftMax layer, which uses a SoftMax function as an

activation function [5]. In this work, we consider three CNN architectures. The first is the basic CNN, the second is the AlexNet and the third is the Xception.

2.1 Basic CNN

The proposed basic CNN architecture is illustrated in Figure 1. It is formed by a successive independent layer of the convolution layer and subsampling layer. Theses layers are followed by an FC layer.

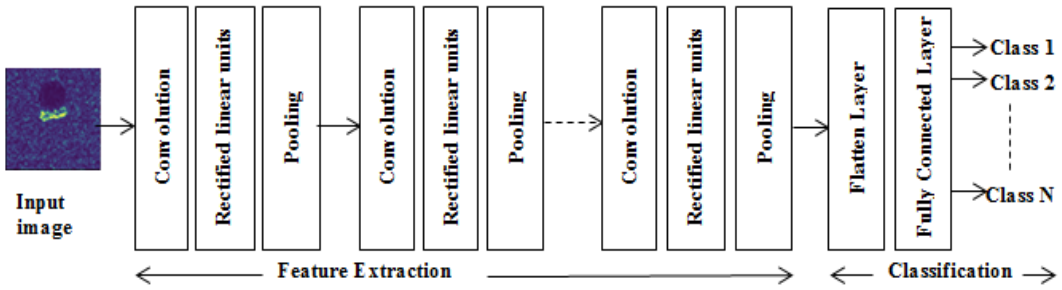


Figure 1. Architecture of a basic CNN model.

To optimize the training process, transfer learning is adopted from the classification of the ImageNet as a source domain. Transfer learning is commonly used in learning applications in order to optimize the learning process and improve the performance of recognition tasks. For the common public architectures, a pre-trained network is used as a starting learning point for a destination task. Thus, the number of classes at the Soft-max layer of the pre-trained CNN network is adjusted.

2.2 AlexNet

AlexNet is a pivotal pre-trained CNN model introduced by Alex Krizhevsky in 2012 [30]. It played a crucial role in advancing computer vision by learning hierarchical representations from 224x224 pixel RGB images. This deep architecture includes convolutional and max-pooling layers for feature extraction, starting from basic patterns and progressing to higher-level abstractions. Extracted features are flattened and processed through three FC layers to facilitate classification; with the final layer having units corresponding to dataset classes and using Softmax activation for probability outputs. In this study, we adapt the structure of the pre-trained network AlexNet, so that it can classify SAR images composed of 10 classes. We therefore act on the FC layer to keep only 10 neurons for the output layers.

2.3 Xception

Xception, derived from "Extreme Inception" is a DCNN (Deep CNN) architecture inspired by inception. It excels at capturing features of different sizes by isolating the learning of spatial and channel-wise features [31]. Instead of combining these types of learning in a single convolution, Xception uses depthwise separable convolutions. It starts with spatial convolutions on each channel and then integrates information across channels using 1x1 convolutions. This design optimizes efficiency while maintaining depth, enabling Xception to efficiently capture both local and global features. It ends with FC layers for feature relationships and uses Softmax activation for multi-class classification, making it highly efficient and accurate for computer vision tasks like image classification. That is why in our study we adapt the output layer to align with the number of classes in the MSTAR dataset.

3. DATA FUSION

To obtain information that is as reliable and precise as possible in all observation conditions, by taking advantage of the complementarity and redundancy of elementary information, as shown in Figure 2, data fusion is used to improve the results. For the data-fusion rule of the different classifiers, the simplest and most popular approach is the majority-vote rule [26]. We will also explore the possibility of applying NNs for the fusion of data from different classifiers. It should be noted that, unlike the

majority rule which aggregates the different elementary decisions to reach the final decision, NNs are instead fed by the distinct outputs of the fully connected layer from each classifier. These outputs represent the results of the Softmax function, effectively providing us with the probabilities for each class. This approach is employed to construct a capable fusion rule, leveraging the capabilities of NNs to model any rule through learning.

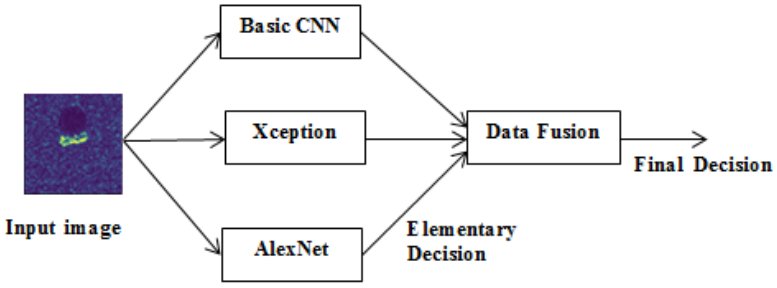


Figure 2. Classification by fusion rule.

3.1 Majority Vote

The majority vote consists in choosing the decision taken by the maximum number of methods. If there are m monitoring methods $\{S_1, S_2, \ldots S_m\}$ such that the method $S_{j \in \{1\ldots m\}}$ assigns the decision D_i to the observation x, noted $S_j(x) = i$, where' i' is the class number and associates an indicator function M_j to each method, such as:

$$M_j^i(x) = \begin{cases} 1 & if \quad S_j(x) = i \\ 0 & else \end{cases} \tag{1}$$

The data fusion from 'm' classifiers is carried out according to the following expression:

$$M_K(x) = \sum_{j=1}^{m} M_K^j \tag{2}$$

The majority-vote rule consists, therefore, in choosing the maximum of M_K. The correct class is the one that is most often chosen by the majority of the methods.

3.2 Rule Based on NNs

Exploring the possibilities offered by NNs for constructing classification rules through learning is a central challenge in the field of image classification. In this study, our focus lies in the classification of SAR images using three distinct approaches. The first involves a basic CNN classifier, while the other two leverage pre-trained architectures; namely, the Xception and AlexNet, renowned for their ability to capture high-level features. A critical step in this methodology involves the use of the Softmax output from each classifier to feed an MLP (Multi-layer Perceptron) NN. This approach skillfully combines the strengths of each classifier while minimizing potential errors. The Softmax function takes as input a vector of scores *S*, where each element S_i represents the degree of belongingness of an example to a particular class. The output of the Softmax function, denoted as *P*, is also a vector of the same dimension as *S*, but each element P_i represents the probability of the example belonging to class *i*.
The mathematical formula for the Softmax function applied to a score vector *S* is as follows:

$$P_i = \frac{e^{S_i}}{\sum_{j=1}^{N} e^{S_j}} \tag{3}$$

where, P_i is the probability that the example belongs to class *i*, S_i the score associated with class *i* and M the total number of classes.

In our approach, we have chosen to utilize the Softmax outputs generated by the classifiers (Basic, AlexNet, Xception) in place of class labels. This decision was made to provide the MLP network with richer and more informative data. Consequently, in our approach, our training dataset comprises input-output vectors. The input vector is composed of the Softmax outputs from each classifier:

$$X = [P_{1,1}\, P_{1,2} \ldots . P_{1,10}\, P_{2,1} P_{2,2} \ldots . P_{2,10}\, P_{3,1} P_{3,2} \ldots\ldots P_{3,10}] \tag{4}$$

where, $P_{i,j}$ is the output of the Softmax function of classifier i for class j, with i ranging from 1 to 3 and j from 1 to 10.

The output vector Y represents the actual outputs of the classes in the MSTAR training dataset, which represents a vector of 10 numerical values, with one value equal to 1, corresponding to the true class, while the others are equal to 0.

One of the learning algorithms for the MLP network is the BP (Backpropagation) algorithm, which is one of the supervised learning algorithms. The principle of this algorithm is based on modifying synaptic weights by propagating the error from the output layer back to the input layer through the intermediate layers.

Therefore, if we consider an MLP network with three layers, the BP algorithm can be listed as follows:

Step 1 - Initialization

We start by initializing the weights for the input layer to the hidden layer in matrix W_1 and the weights from the hidden layer to the output layer in matrix W_2. We also initialize the biases in the vectors b_1 and b_2.

Step 2 - Forward Propagation

We calculate the output of the neurons in the hidden layer using the sigmoid activation function:

$$Z_1 = X.W_1 + b_1 \tag{5}$$

$$A_1 = \frac{1}{1+e^{-Z1}} \tag{6}$$

We calculate the output of the neurons in the output layer using a linear activation function:

$$Z_2 = A_1.W_2 + b_2 \tag{7}$$

$$A_2 = Z_2 \tag{8}$$

Step 3 - Prediction

The output A_2 contains predictions for the 10 classes.

Step 4 - Error Calculation

We calculate the error by comparing predictions A_2 with the true values Y, using the MSE (Mean Squared Error), as follows:

$$Erreur = MSE(A2, Y) \tag{9}$$

Step 5 - Backpropagation

We use backpropagation to calculate the gradients of the error with respect to weights W_1 and W_2, as well as biases b_1 and b_2.

$$\nabla W_1 = \frac{\partial Erreur}{\partial W_1} \tag{10}$$

$$\nabla W_2 = \frac{\partial Erreur}{\partial W_2} \tag{11}$$

$$\nabla b_1 = \frac{\partial Erreur}{\partial b_1} \tag{12}$$

$$\nabla b_2 = \frac{\partial Erreur}{\partial b_2} \tag{13}$$

Step 6 - Weight Update

We use the calculated gradients to update the network's weights

$$W_1 = W_1 - \eta \nabla W_1 \tag{14}$$

$$W_2 = W_2 - \eta \nabla W_2 \tag{15}$$

$$b_1 = b_1 - \eta \nabla b_1 \tag{16}$$

$$b_2 = b_2 - \eta \nabla b_2 \tag{17}$$

Each time, we present a new input vector to the network with its associated output and repeat the calculation process from Step 2. Once we have presented all the examples in the training dataset, we calculate the sum of all the errors.

$$E = \sum_{K=1}^{M} Erreur_K \quad (18)$$

where, M is the total number of examples in the training dataset.

If we reach the desired error, the learning process ends. Otherwise, we repeat the algorithm from Step 2 with all the examples in the training dataset.

To make the learning algorithm faster, a term called Momentum, which takes into account the weight changes between two successive iterations, is added to Equations 14-17.

Thus, our goal is to improve the overall performance of SAR image classification, by judiciously combining the results of different classifiers, using the power of an MLP network. Subsequently, this MLP network will be trained on the MSTAR training dataset. The architecture chosen will be the one that gives the best performance in terms of correct-classification rates, obtained through meticulous tuning of hyperparameters to achieve an optimal configuration.

4. RESULTS AND DISCUSSION

In this work, we classified the SAR images from the MSTAR database, according to 10 class categories. In other words, we investigated the different architectures of DL for the classification of SAR satellite images. That is, we started with a simple architecture (Basic CNN) in which we looked for the optimal one. Next, based on the concept of transfer learning, we examined the pre-training architectures to use them in this work. Subsequently, we integrated the outcomes from the three CNNs through a dual approach. The initial method involves the use of the majority fusion rule, while the second method employs the capabilities of NNs to learn and implement the fusion rule *via* supervised learning. To begin, we introduce the software and hardware tools employed in our study. Subsequently, we detail the dataset utilized in this paper. Lastly, we assess the performance of the individual architectures, along with their combined counterpart, using the confusion matrix as a key reference. Specifically, we will focus on the accurate classification rate

4.1 Software and Hardware Tools

To conduct our experiments and effectively categorize all SAR images, we employed the Matlab software due to its user-friendly programming environment and extensive repository of image processing and ML resources. This encompasses artificial NNs and DL capabilities. For the DL framework, we utilized the DL toolbox. It is worth noting that the version of CUDA used in our experiments is 11.2. This combination allowed us to efficiently implement and train our CNN models, including the Basic CNN, earning AlexNet and Xception architectures.

Regarding the MLP network, we employed the NN toolbox to design and implement it. This approach provided the necessary tools for our network's development and evaluation; contributing to the overall success of our methodology.

Regarding our hardware setup, the configuration employed for training and assessing CNN models consisted of an Intel® Core i7 microprocessor running at 3.5GHz, coupled with 64GB of RAM. Additionally, a NVIDIA RTX 2080 TI GPU has been integrated into the system.

4.2 Database of SAR Images

To implement the proposed classification methodology utilizing various DL techniques, the MSTAR database of SAR images was employed. Widely utilized in the literature due to its public availability, the MSTAR dataset encompasses a collection of SAR images with a resolution of 0.3 m × 0.3 m, captured using an X-band spotlight SAR sensor [29].

Figure 3 illustrates the MSTAR database, housing SAR images of military vehicles and organized into 10 distinct class categories. Within the dataset, a total of 5165 images exist, with 2740 images allocated for training and 2425 images designated for the test dataset. The distribution of images across each target (class) can be found in Table 1. These images depict ten distinct classes of ground targets, encompassing entities, like tanks (T62 and T72), rocket launchers (2S1), trucks (ZIL131), armored personnel carriers (BTR70, BTR60, BRDM2 and BMP2), air-defense units (ZSU23/4) and bulldozers (D7). Notably, the images were captured under diverse conditions, spanning different

aspect angles, depression angles and serial numbers. To evaluate the effectiveness of our approach, the MSTAR base, captured under SOC (Standard Operating Conditions), was considered [29].

Table 1. Training and testing MSTAR dataset.

Targets	Train	Test
2s1	299	274
BMP2	233	195
BRDM2	298	274
BTR60	256	195
BTR70	233	196
D7	299	274
T62	299	273
T72	232	196
ZILI131	299	274
ZSU234	299	274

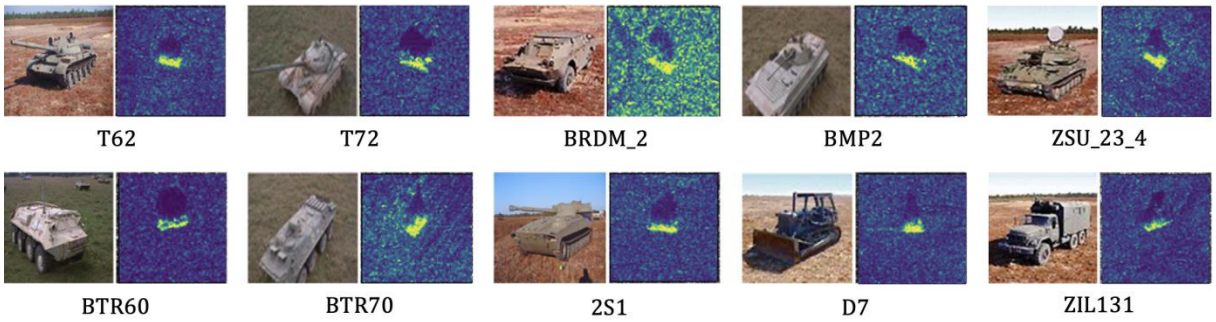


Figure 3. Military vehicles from MSTAR database and their corresponding optical images.

4.3 Simulation Results for Basic CNN

Based on the basic-CNN architecture, the proposed approach determines the number of convolution layers, the number of filters, the size of the convolution filters as well as the subsampling step. As this is an optimization problem the inherent CNN hyper-parameters of which are difficult to get, we remedy it by giving initial values for all CNN hyper-parameters. Therefore, the training stage is to change a single hyper-parameter until the optimal value yields a better classification. The process is repeated for the other hyper-parameters until all optimal hyper-parameters are obtained.

Here, we used 158×158 input images and proposed a basic CNN architecture with initial hyper-parameters as given in Table 2.

Table 2. Initial hyper-parameter values.

Hyper-parameters	
Number of convolutional layers	4
Size of filters (stride=2)	3x3
Number of filters	5
Size of max-pooling (stride=2)	5x5
Size of epoch	5
Initial learning rate	10^{-1}
Percentage of the learning/validation database	0.5

Note that these hyper-parameters were obtained separately from each other. Therefore, this allowed us to locate the range of hyper-parameters to be investigated. After successive tests, we found the optimal hyper-parameters, shown in Table 3. These latter, which allowed us to achieve the best performances, are discussed in the next sub-section.

Table 3. Optimal hyper-parameters.

Optimal hyper-parameters	
convolutional layers Number	3
Size of filters (stride=2)	7x7
Filters number	16
Size of max-pooling (stride=2)	2x2
Epoch number	10^3
Initial learning rate	10^{-2}
Percentage of the learning/validation database	0.9

Table 4 shows the performance evaluation of the basic CNN in terms of correct classification rate on the basis of SAR imaging test (MSTAR). The results were obtained through the training of the basic CNN, with the optimal hyper-parameters, achieving an overall correct-classification rate of 97.3%. It should also be noted that for the ZSU234 military machine, the correct-classification rate reached 100%.

Table 4. Confusion matrix of the proposed basic CNN model.

Classes	**2S1**	**BMP 2**	**BRDM2**	**BTR60**	**BTR70**	**D7**	**T62**	**T72**	**ZILI131**	**ZSU234**	**RR%**
2S1	263	0	0	0	0	0	0	0	1	10	95.99
BMP 2	9	167	2	6	3	0	0	7	1	0	85.64
BRDM2	0	0	269	0	0	0	0	4	1	0	98.18
BTR60	6	1	5	181	2	0	0	0	0	0	92.82
BTR70	0	0	1	2	193	0	0	0	0	0	98.47
D7	0	0	2	0	0	271	0	0	1	0	98.91
T62	2	0	0	0	0	0	269	0	2	0	98.53
T72	1	1	1	1	0	0	0	192	0	0	97.96
ZILI131	1	0	0	0	0	0	0	0	273	0	99.64
ZSU234	0	0	0	0	0	0	0	0	0	274	100.00
Average Accuracy = 96.61											

In order to determine the architecture that yields a good classification of the SAR images, the next sub-section is devoted to the investigation of the two other pre-trained deep-learning architectures.

4.4 Simulation Results for the AlexNet and Xception Networks

AlexNet and Xception are transfer-learning networks that are based on pre-trained models. The model that we used has already been previously presented.

To classify new SAR images, we recycle a pre-trained network by replacing the three final layers with new layers adapted to the new dataset. Here, we need to change the number of classes to 10 and retain the parameters for the other layers.

Finally, a training operation is launched, so that the new network adapts with the new learning data. For a better classification, this is accomplished through the adjustment of the network parameters.

Table 5. Optimal hyper-parameters for AlexNet.

Optimal hyper-parameters	
MiniBatchSize	18
ValFrequency	10
Maximum epoch	70
Initial learning rate	1e-2
Percentage of the learning/validation database	0.6

Table 6. Optimal hyper-parameters for Xception.

Optimal hyper-parameters	
MiniBatchSize	36
ValFrequency	32
Maximum epoch	30
Initial learning rate	1e-4
Percentage of the learning/validation database	0.9

It should be noted that the optimal hyper-parameters of the two pre-trained networks were found by successive tests. Tables 5 and 6 show the optimal hyper-parameters that were obtained via their

adjustment, element by element.

Simulation results on the testing MSTAR dataset are shown in Tables 7 and 8.

Table 7. Confusion matrix of AlexNet.

Classes	2S1	BMP 2	BRDM2	BTR60	BTR70	D7	T62	T72	ZILI131	ZSU234	RR%
2S1	215	0	4	2	0	0	50	0	0	3	78.47
BMP 2	3	175	2	2	0	0	0	13	0	0	89.74
BRDM2	0	0	274	0	0	0	0	0	0	0	100.00
BTR60	0	0	0	191	0	0	0	4	0	0	97.95
BTR70	0	1	0	4	191	0	0	0	0	0	97.45
D7	1	0	0	0	0	266	2	0	0	5	97.08
T62	1	0	0	0	0	0	251	0	0	21	91.94
T72	0	0	1	1	0	0	0	194	0	0	98.98
ZILI131	0	0	0	0	0	0	0	0	274	0	100.00
ZSU234	0	0	1	0	0	0	0	0	0	273	99.64
Average Accuracy=95.12											

Table 8. Confusion matrix of Xception.

Classes	2S1	BMP 2	BRDM2	BTR60	BTR70	D7	T62	T72	ZILI131	ZSU234	RR%
2S1	269	0	1	0	0	0	1	0	0	3	98.18
BMP 2	0	178	3	2	1	0	0	11	0	0	91.28
BRDM2	0	0	273	1	0	0	0	0	0	0	99.64
BTR60	0	0	0	193	2	0	0	0	0	0	98.97
BTR70	0	0	0	0	196	0	0	0	0	0	100.00
D7	3	0	0	0	0	266	0	0	1	4	97.08
T62	1	0	0	0	0	0	271	0	0	1	99.27
T72	0	0	0	1	0	0	0	195	0	0	99.49
ZILI131	0	0	0	0	0	0	0	0	274	0	100.00
ZSU234	0	0	0	0	0	0	0	0	0	274	100.00
Averarage Accuracy=98.39											

Finally, based on Tables 4, 7 and 8, we achieved better performance with the Xception classifier, achieving a correct-classification rate of 98.39%. This was followed by the Basic classifier, which achieved a rate of 96.61% and lastly came the AlexNet classifier, which achieved a classification rate of approximately 95.12%.

After having trained several DL architectures for the classification of SAR images, it is important to consider the possibility of exploring the theory of data fusion for the classification of SAR images. To achieve this, we use a parallel architecture, in which we apply the fusion rule via the majority vote as well as the use of NNs for the construction of a new data-fusion rule.

4.5 Simulation Results with Decision Fusion Techniques

In this sub-section, we use a parallel architecture consisting of three classifiers; namely, the basic CNN, the two pre-trained networks, AlexNet and Xception, which are passed to the fusion center. In the following part, we will evaluate the performance of the two data fusion approaches; namely, the majority rule and the rule based on NNs. The two techniques will be evaluated *via* the MSTAR testing dataset.

4.5.1 Majority Voting Rule

In this technique, the basic decisions from each classifier are transmitted to the fusion center for a final determination using a majority voting rule for data fusion. It is worth noting that in instances where the three classifiers yield differing classes, the rule will yield a random decision. The performance of this architecture was appraised using the MSTAR test database. The resulting confusion matrix is detailed in Table 9, showing an average global accuracy rate of 99.53%; indicating a gain of 1.14%.

4.5.2 Neural Network

The second technique which we used is data fusion using ANNs (Artificial NNs), exploiting the ability of NNs to model any rule by the supervised learning. Therefore, we investigated several architectures of NNs; the MLP architecture has been retained. In our investigation, we used the Levenberg-Marquardt learning algorithm to train the network. Levenberg-Marquardt algorithm is a widely used

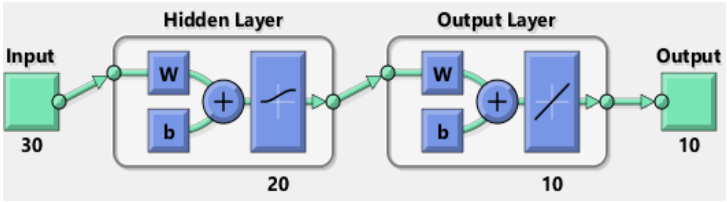
optimization algorithm for training NNs. It adjusts the network's weights to minimize the error between the predictions and the actual values.

Table 9. Confusion matrix of majority-voting rule.

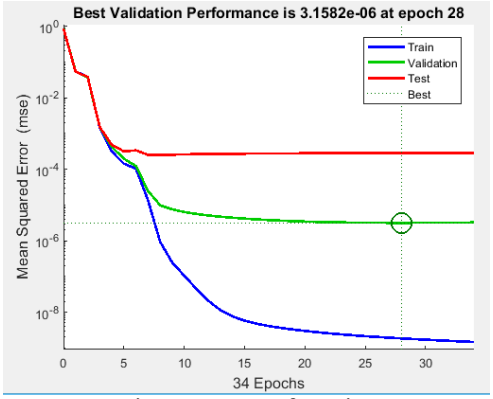
Classes	2S1	BMP 2	BRDM2	BTR60	BTR70	D7	T62	T72	ZILI131	ZSU234	RR%
2S1	272	0	0	0	0	0	0	0	0	2	99.27
BMP 2	0	189	1	1	0	0	0	4	0	0	96.92
BRDM2	0	0	274	0	0	0	0	0	0	0	100.00
BTR60	0	0	0	195	0	0	0	0	0	0	100.00
BTR70	0	0	0	0	196	0	0	0	0	0	100.00
D7	0	0	0	0	0	274	0	0	0	0	100.00
T62	0	0	0	0	0	0	272	0	0	1	99.63
T72	0	0	0	1	0	0	0	195	0	0	99.49
ZILI131	0	0	0	0	0	0	0	0	274	0	100.00
ZSU234	0	0	0	0	0	0	0	0	0	274	100.00
Average Accuracy=99.53											

The configuration of this network was iteratively fine-tuned through multiple experiments, involving the minimization of a cost function and adjustments of hyper-parameters, such as the number of hidden layers, the number of neurons in the hidden layers, the activation functions, the error metrics and the Momentum parameter, using the MSAR training dataset. This iterative process continued until achieving optimality, as demonstrated in Figures 4 and 5, which illustrate the optimal structure of the MLP network and the loss function.

Upon completing the training phase, simulation results on the MSAR test dataset yielded an average overall accuracy rate of 99.44%, as depicted in the confusion matrix of Table 10, indicating a significant improvement.



Figue 4. NN-based data-fusion center.



Figue 5. Loss function.

Table 10. Confusion matrix of NN-based data-fusion center.

Classes	2S1	BMP 2	BRDM2	BTR60	BTR70	D7	T62	T72	ZILI131	ZSU234	RR%
2S1	272	0	0	0	0	0	0	0	0	2	99.27
BMP 2	0	191	0	1	0	0	0	3	0	0	97.95
BRDM2	0	0	274	0	0	0	0	0	0	0	100.00
BTR60	2	0	0	193	0	0	0	0	0	0	98.97
BTR70	0	0	0	1	195	0	0	0	0	0	99.49
D7	1	0	0	0	0	273	0	0	0	0	99.64
T62	0	0	0	0	0	0	272	0	0	1	99.63
T72	0	0	0	1	0	0	0	195	0	0	99.49
ZILI131	0	0	0	0	0	0	0	0	274	0	100.00
ZSU234	0	0	0	0	0	0	0	0	0	274	100.00
Average Accuracy=99.44											

4.6 Execution-time Analysis for the Proposed Methods

We have taken into consideration the execution time of the two proposed methods, recognizing the crucial importance of speed in radar detection and SAR image-classification applications, which often operate in real time. Firstly, it is important to note that the three used DL architectures require significant time during the training phase. However, once this phase is completed "in offline" mode, we measured the execution time of the three architecture; namely, the basic CNN, the AlexNet and the

Xception. The execution times that we obtained for classifying a single image are 0.908 ms, 2.76 ms and 11.48 ms, respectively.

It is essential to note that these execution times are calculated for classifying a single image. In a parallel implementation, we consider the longest time among these three times, which is that of the Xception architecture, as the next step can only occur when the three outputs from the three classifiers are ready.

Regarding the execution time for data fusion, we observed that using the majority rule requires approximately 15.8 µs, while an MLP network requires about 20.86 µs. It is worth noting that these times are practically comparable, with a slight advantage for the majority rule.

In summary, while the three used DL architectures require a significant amount of time during the training phase, once this phase is completed, our fusion methods present execution times that are compatible with the requirements of real-time radar detection and SAR image-classification applications; thus demonstrating their practicality and efficiency.

4.7 Performance Comparison of Different Methods

In order to facilitate the comparison of our findings with other DL methodologies assessed on the MSATR test dataset, we present a summary of methods and their corresponding average classification rates in Table 11. Particularly, our approach, employing a parallel architecture with a fusion center, achieves the highest average accuracy in correct classification. This outcome underscores how our strategy of data fusion effectively leverages the unique strengths of individual classifiers while mitigating classification errors. Note that our majority voting-based fusion rule exhibits a superior performance. Additionally, it is worth highlighting that the data-fusion technique centered on ANNs also delivers highly commendable results.

Table 11. Performance comparison of different methods

METHOD	AVERAGE ACCURACY (%)
FCNN [33]	98.10
ENHANCED-SHAPE CNN [34]	99.29
MFFD-CNN [35]	99.30
FEF-NET [36]	99.31
CNN & SVM [37]	99.50
PROPOSED METHOD	99.53

5. CONCLUSION

The primary goal of this study was to employ a combined architecture consisting of three classifiers and a data-fusion center for the purpose of classifying SAR satellite images. To achieve this, a basic CNN architecture was utilized. This architecture underwent optimization by fine-tuning the network parameters, including hyper-parameters, until reaching the optimal configuration. In addition to this, two other pre-trained CNN networks were employed. For these networks, the final three layers were adapted to facilitate SAR image classification. A similar optimization process was applied to determine the best architecture. To merge the outputs of the distinct classifiers, a fusion center was implemented. Two techniques were employed for this fusion; namely, the majority-voting rule and the utilization of NNs to construct a learning decision rule. Through simulations conducted on SAR images, with a focus on the MSTAR database, the effectiveness of data fusion in SAR image classification was demonstrated. Notably, the majority fusion rule technique exhibited an exceptional performance, achieving an accurate-classification rate of 99.53%. As future works and in order to validate the performance of the proposed approach, extensive experiments on other datasets should be implemented with other fusion techniques.

REFERENCES

[1] L. M. Novak, "State-of-the-art of SAR Automatic Target Recognition," Proc. of Record of the IEEE 2000 Int. Radar Conf., pp. 836-843, Alexandria, VA, USA, 2000.

[2] A. Karine, A. Toumi, A. Khenchaf and M. E. Hassouni, "Saliency Attention and Aift Key-points Combination for Automatic Target Recognition on MSTAR Dataset," Proc. of the Int. Conf. on

Advanced Technologies for Signal and Image Processing (ATSIP), pp. 1-5, Fez, Morocco, 2017.

[3] D. R. Hush and B. G. Horne, "Progress in Supervised Neural Networks," IEEE Signal Processing Magazine, vol. 10, no. 1, pp. 8-39, 1993.

[4] T. D. Ross et al., "Standard SAR ATR Evaluation Experiments Using the MSTAR Public Release Data Set," Proc. SPIE, Algorithms for Synthetic Aperture Radar Imagery V, vol. 3370, pp.566–573, 1998.

[5] A. Toumi and A. Khenchaf, "Aircrafts Recognition Using Convolutional Neurons Network," Proc. of the Int. Conf. on Radar Systems (RADAR'17), pp.23-26, UK, 2017.

[6] A. Masoomi, R. Hamzehyan and N.C. Shirazi, "Speckle Reduction Approach for SAR Image in Satellite Communication," Int. Journal of Machine Learning and Computing, vol. 2, no. 1, 2012.

[7] A. Toumi, A. Alhousseini and A. Khenchef, "Aircraft Recognition Using Convolutional Neurons Network," Proc. of the IEEE 7th Seminar on Detection Systems: Architectures and Technologies (DAT 2017), Algeria, 20-22 February 2017.

[8] Q. Zhao and J. C. Principe, "Support Vector Machines for SAR Automatic Target Recognition," IEEE Transactions on Aerospace and Electronic Systems, vol. 37, pp. 643–654, 2001.

[9] Y. Sun, Z. Liu, S. Todorovic and J. Li, "Adaptive Boosting for SAR Automatic Target Recognition," IEEE Transactions on Aerospace and Electronic Systems, vol. 43, no.1, pp. 112–125, 2007.

[10] A. Karine, A. Toumi, A. Khenchaf and M. Hassouni, "Multivariate Copula Statistical Model and Weighted Sparse Classification for Radar Image Target Recognition," Computers & Electrical Engineering, vol. 84, p. 106633, 2020.

[11] J. C. Cexus, A. Toumi and M. Riahi, "Target Recognition from ISAR Image Using Polar Mapping and Shape Matrix," Proc. of the 2020 IEEE 5th Int. Conf. on Advanced Technologies for Signal and Image Processing (ATSIP), pp. 1-6, Sousse, Tunisia, September 2020.

[12] C. Coman and R. Thaens, "A Deep Learning SAR Target Classification Experiment on MSTAR Dataset," Proc. of the 19th IEEE Int. Radar Symposium (IRS 2018), Bonn, Germany, 20-22 June 2018.

[13] S. A. Wagner, "SAR ATR by a Combination of Convolutional Neural Network and Support Vector Machines," IEEE Trans. on Aerospace and Electronic Systems, vol. 52, no. 6, pp. 2861–2872, 2016.

[14] A. David and E. Morgan "Deep Convolutional Neural Networks for ATR from SAR Imagery," Proc. of SPIE, vol. 9475, DOI: 10.1117/12.2176558, 2015.

[15] A. Ameta, V. Singh1 and V. Devi, "A Convolutional Neural Network Based Approach for SAR Image Classification of Vehicles," Int. Journal of Engineering Research & Technology (IJERT), vol. 9, no. 06, pp. 544-547, 2020.

[16] A. T. Al-Taani and I. T. Al-Dagamesh, " Automatic Detection of Pneumonia Using Concatenated Convolutional Neural Network", Jordanian Journal of Computers and Information Technology (JJICIT), vol. 09, no. 02, pp. 118-136, 2023.

[17] R. Hang, Q. Liu, D. Hong and P. Ghamisi, "Cascaded Recurrent Neural Networks for Hyper-spectral Image Classification," IEEE Trans. Geoscience and Remote Sensing, vol. 57, pp. 5384–5394, 2019.

[18] C. Wang, X. Liu, J. Pei, Y. Huang, Y. Zhang and J. Yang, "Multi-view Attention CNN-LSTM Network for SAR Automatic Target Recognition," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 14, pp. 12504-12513, 2021.

[19] X. Fan, L. Chen, X. Xu, C. Yan, J. Fan and X. Li, "Land Cover Classification of Remote Sensing Images Based on Hierarchical Convolutional Recurrent Neural Network," Forests, vol. 14, no. 9, p. 1881, 2023.

[20] Z. Huang, Z. Pan and B. Lei, "Transfer Learning with Deep Convolutional Neural Network for SAR Target Classification with Limited Labeled Data," Remote Sensing, vol. 9, no. 9, p. 907, 2017.

[21] Y. Zhai, H. Ma et al., "SAR ATR with Full-angle Data Augmentation and Feature Polymerization," IET, vol. 19, pp. 6226-6230, 2019.

[22] Z. Geng, Y. Xu, B.-N. Wang, X. Yu, D.-Y. Zhu and G. Zhang, "Target Recognition in SAR Images by Deep Learning with Training Data Augmentation" Sensors, vol. 23, no. 2, p. 941, 2023.

[23] J. Ding, B. Chen, H. Liu and M. Huang, "Convolutional Neural Network with Data Augmentation for SAR Target Recognition," IEEE Geoscience and Remote Sensing Letters, vol. 13, no.3, pp. 364–368, 2016.

[24] L. Lin, J. Li, Q. Yuan and H. Shen, "Polarimetric SAR Image Super-resolution *via* Deep Convolutional Neural Network", Proc. of the IEEE Int. Geoscience and Remote Sensing Symposium (IGARSS 2019), DOI: 10.1109/IGARSS.2019.8898160, Yokohama, Japan, 2019.

[25] Z. Cui, M. Zhang, Z. Cao and C. Cao, "Image Data Augmentation for SAR Sensor *via* Generative Adversarial Nets," IEEE Access, vol. 7, pp. 42255 – 42268, 2019.

[26] G. Joshi, R. Natsuaki and A. Hirose "Multi-sensor Satellite-imaging Data Fusion for Earthquake Damage Assessment and the Significant Features," Proc. of the 7th Asia-Pacific Conf. on Synthetic Aperture Radar (APSAR), DOI: 10.1109/APSAR52370.2021.9688438, 2021.

[27] Y. Khenchaf and A. Toumi, "Siamese Neural Network for Automatic Target Recognition Using Synthetic Aperture Radar," Proc. of the Int. Geoscience and Remote Sensing Symposium (IGARSS), Pasadena, USA, Jully 2023.

[28] C. Zhang, Y. Wang, H. Liu, Y. Sun and L. Hu, " SAR Target Recognition Using Only Simulated Data for Training by Hierarchically Combining CNN and Image Similarity," IEEE Geoscience and Remote Sensing Letters, vol. 19, pp. 1-5, 2022.
[29] Master Public Targets, The Sensor Data Management System, [Online], Available: https://www.sdms.afrl.af.mil/index.php?collection=mstar&page=targets.
[30] A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems (NIPS), pp. 1-9, 2012.
[31] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1251-1258, 2017.
[32] G. Niu, S. S. Lee, B. S. Yang and S. J. Lee, "Decision Fusion System for Fault Diagnosis of Elevator Traction Machine," Journal of Mechanical Science and Technology, vol. 22, no.1, pp. 85-95, 2008.
[33] L. Yu et al., "SAR ATR Based on FCNN and ICAE," Journal of Radars, vol. 7, no.5, pp. 622-631, 2018.
[34] M. Huang, F. Liu and X. Meng, "Few Samples of SAR Automatic Target Recognition Based on Enhanced-shape CNN," Journal of Mathematics, vol. 2021, pp.1-16, 2021.
[35] L. Guo, "SAR Image Classification Based on Multi-feature Fusion Decision Convolutional Neural Network," IET Image Process, vol. 16, pp.1–10, 2022.
[36] J. Pei, Z. Wang et al., "FEF-Net: A Deep Learning Approach to Multi-view SAR Image Target Recognition, " Remote Sensing, vol. 13, no.17, p. 3493, 2021.
[37] S. A. Wagner, "SAR ATR by a Combination of Convolutional Neural Network and Support Vector Machines," IEEE Transactions on Aerospace and Electronic Systems, vol. 52, no. 6, pp. 2861-2872, 2016.

ملخص البحث:

في تطبيقاتٍ متعدّدة للتّصوير بالرّادار، ترتبط إحدى المشكلات الأساسية بتحليل الصُّور وتفسيرِها، وخصوصاً عند تمييز الأهداف المتحركة والثابتة. وقد ازداد هذا الأمر صعوبةً بسبب الكمّ الهائل من البيانات، وأدّى ذلك الى استخدام طرقٍ أوتوماتيكية لمعالجة البيانات وتمييز الأهداف.

يهدف هذا البحث الى استقصاء دمْج البيانات في مصنّفات صُور الرّادار، وفيه نقترح طريقةً جديدة لدمْج ثلاثٍ من بِنى الشّبكات العصبية الالتفافية، وهي الشّبكة العصبية الالتفافية الأساسية، وبِنية (Xception)، وبِنية (AlexNet). ويتمّ ذلك باستخدام طريقتين من طُرق الدّمْج، تعتمد أولاهما على قاعدة انتخاب الأغلبية، بينما تستخدم الثانية شبكةً عصبية لدمج القرارات من ثلاثة مصنِّفات أولية للوصول الى القرار النّهائي.

ولتقييم الطّريقة المقترحة، تمّ تجريبها على مجموعة بيانات (MSTAR) الخاصّة بكشف وتمييز أهداف الرّادار المتحركة والثابتة. وقد ادّى استخدام تقنية الدّمج المقترحة في هذا البحث الى تحسين معدّل التّمييز بدّقة نهائية بلغت 99.59% لطريقة الدّمج الخاصّة بقاعدة انتخاب الأغلبية و 99.51% لطريقة الدّمج باستخدام الشّبكة العصبية، حيث تجاوزت هذه المعدّلات تلك التي تمّ الحصول عليها باستخدام شبكة عصبية التفافية مفرَدة.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

A NOVEL APPROACH TO INTRUSION-DETECTION SYSTEM: COMBINING LSTM AND THE SNAKE ALGORITHM

Sanaa Ali Jabber[1], Soukaena H. Hashem[2] and Shatha H. Jafer[2]

(Received: 7-Sep.-2023, Revised: 29-Oct.-2023, Accepted: 11-Nov.-2023)

ABSTRACT

In the epoch of digital transformation, cloud computing remains paramount, acting as the linchpin for a plethora of services from enterprise solutions to day-to-day consumer applications. Yet, its expansive nature has invariably rendered it susceptible to a myriad of cyber threats, necessitating advanced, adaptive defense mechanisms. This paper introduces a novel intrusion-detection method tailored for cloud environments, ingeniously amalgamating the temporal pattern-recognition capabilities of Long Short-Term Memory (LSTM) networks with the heuristic finesse of the Snake algorithm. Our research meticulously delineates the LSTM-Snake model's design, implementation and exhaustive benchmarking against prevailing approaches for a rigorous and comprehensive evaluation of cloud-based intrusion-detection systems and by using the TON-IOT dataset, a carefully curated dataset tailored for cloud-centric applications. The experimental results underscore the model's prowess, registering a commendable 99% accuracy rate in intrusion detection; a marked improvement over current state-of-the-art methodologies. The ensuing discussions offer insights into the model's practical implications and potential limitations.

KEYWORDS

Cyber threats, Intrusion detection, Cloud environments, Long short-term memory (LSTM), Snake algorithm, Intrusion detection systems (IDSs).

1. INTRODUCTION

In the ever-evolving landscape of cyber threats, securing cloud-computing infrastructure remains paramount [1]. The proliferation of cloud technologies has yielded remarkable efficiencies and scalability in the computational realm, but it has concurrently expanded the surface for potential cyber attacks. Intrusion Detection Systems (IDSs) have traditionally been employed to monitor and counteract these malicious activities, but as the complexity and stealth of attacks have evolved, so has the need for more advanced detection methodologies [2]. Deep learning, a subdomain of machine Learning, has shown immense promise in numerous applications ranging from computer vision to natural-language processing [3]. Notably, Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), have demonstrated proficiency in processing and predicting sequences, which makes them particularly suitable for time-series data, like network traffic [4]. The rationale for integrating deep learning into IDS stems from its ability to discern intricate patterns in vast datasets, potentially uncovering novel attack vectors that traditional methods might overlook. However, like many deep-learning models, LSTMs require meticulous tuning to function optimally, a process often constrained by the vastness of the hyper-parameter space. Traditional methods of optimization can be time-consuming and might not guarantee convergence to the best model parameters [5]. In this context arises the motivation to explore alternative optimization techniques, like the Snake algorithm [6], inspired by the foraging behavior of snakes. By leveraging such bioinspired algorithms, the hope is to enhance the efficiency and efficacy of LSTM-based intrusion detection, ensuring that cloud-computing environments remain resilient against the multifarious cyber threats that persist in the digital age.

The overarching objective of this research is to foster a heightened level of cybersecurity within cloud-computing environments by ingeniously amalgamating the predictive prowess of LSTM networks with the optimization efficiency of the Snake algorithm. Initially, we aim to dissect the

1. S. Ali Jabber is with Department of Faculty of Administration and Economics, AL-Muthanna University, AL-Muthanna, Iraq. Email: sanaa.ali@mu.edu.iq
2. S. H. Hashem and S. H. Jafer are with Department of Computer Science, University of Technology, Baghdad, Iraq. Emails: Soukaena.h.hashem@uotechnology.edu.iq and Shatha.h.jafer@uotechnology.edu.iq

intricacies of modern cyber threats in the cloud realm, laying a solid groundwork for the necessity of advanced intrusion-detection mechanisms. At the crux of our endeavor lies the objective to design, implement and fine-tune an LSTM-based Intrusion Detection System (IDS) that can efficiently process and analyze time-series network-traffic data, uncovering both overt and covert malicious activities. Recognizing the inherent challenges associated with hyper-parameter tuning in deep-learning models, a pivotal goal is to harness the Snake Algorithm's bio-inspired optimization techniques. By doing so, we aim to expedite the search for optimal LSTM configurations while ensuring robustness against a wide array of cyber threats. Comparative analysis forms another essential objective, as we aspire to benchmark our LSTM-Snake Algorithm hybrid against conventional intrusion-detection methods, both in terms of accuracy and computational efficiency. Furthermore, the research intends to assess the real-world applicability of our proposed model, gauging its performance in dynamic cloud environments subjected to contemporary-attack vectors [7].

2. CONTRIBUTIONS

In the multifarious realm of cybersecurity research, our contribution endeavors to pioneer a symbiotic melding of the robustness of Long Short-Term Memory (LSTM) networks with the adaptive finesse of the Snake optimization technique [6]. While LSTMs, with their inherent capability to understand and process sequential data, have been recognized for their potential in intrusion detection, their performance is critically dependent on the precise configuration of their hyper-parameters. Traditional techniques for hyper-parameter tuning often either fall into the trap of computational extravagance or suffer from local-optima stagnation [5]. It is here that our study introduces a seminal innovation. Drawing inspiration from the adaptive, heuristic-based searching capabilities of the Snake optimization technique, we propose a novel method for hyper-parameter tuning that allows for a more dynamic, efficient and expansive exploration of the hyper-parameter space. This synergistic approach not only seeks to enhance the efficiency of LSTM networks in detecting intrusions, but also aims to provide a more generalized and adaptable model that can evolve with the rapidly shifting contours of the IoT threat landscape. In doing so, our work aspires to bridge a significant gap in the current literature, offering a scalable and adaptive solution that marries the strengths of deep learning with the agility of heuristic optimization.

3. LITERATURE REVIEW

3.1 Cybersecurity Threat Landscape in Cloud Computing

In the contemporary era of digital transformation, cloud computing stands at the vanguard, offering businesses and individuals alike unparalleled advantages in scalability, flexibility and cost-efficiency. However, with this paradigm shift to decentralized and virtualized computing resources, there emerges a sophisticated and continually evolving cybersecurity-threat landscape. As organizations migrate their critical data and applications to the cloud, they inadvertently expose themselves to a myriad of vulnerabilities and attack vectors. One of the most pronounced threats in cloud environments is data breaches, where malevolent actors seek unauthorized access to sensitive data, potentially leading to catastrophic financial and reputational ramifications [8]. Misconfigurations, often resulting from the complexity of cloud setups combined with a lack of expertise, have frequently been the Achilles' heel, inadvertently leaving data stores unprotected and accessible [1]. Similarly, inadequate access controls can allow both internal and external threat actors to escalate privileges and misuse resources [9]. The shared responsibility model of cloud security, where both the cloud service provider and the client have delineated security duties, often introduces ambiguities that malicious entities can exploit. Further exacerbating the threat landscape is the increasing prevalence of Distributed Denial of Service (DDoS) attacks targeting cloud infrastructures [10], aiming to disrupt services and potentially camouflage other malicious activities. Additionally, we cannot overlook insecure Application Programming Interfaces (APIs), which, if not meticulously designed and secured, can become gateways for cyberattacks. The advent of cloud-native technologies, such as containerization and serverless computing, while promising, introduces their own set of security challenges that are still in the early stages of understanding. Amid this backdrop, it's palpable that the cloud, while being transformative, has ushered in a complex cybersecurity milieu. Navigating this

landscape requires a thorough comprehension of the threats, an anticipation of emerging risks and a proactive, multi-layered defense strategy.

3.2 Intrusion-detection Techniques for Cloud Computing

Cloud computing, often seen as the heartbeat of modern tech trends, has reshaped the way in which companies work, offering them huge perks, like the ability to scale up or down easily, more flexibility and a cost-effective approach. However, behind this revolutionary framework comes a complicated network of cybersecurity challenges. The need to protect cloud infrastructures has prompted the development and adoption of Intrusion Detection Systems (IDSs) particularly designed for the cloud. Intrusion-detection techniques are roughly classified into two categories: signature-based and anomaly-based approaches. Signature-based approaches, also known as misuse-detection approaches, rely on pre-existing databases of known attack patterns or 'signatures' [11]. While being extremely effective at identifying known threats, its intrinsic drawback is their inability to detect novel or zero-day assaults. This is when anomaly-based approaches come into play.

They intend to discover deviations or abnormalities suggestive of possible intrusions by creating a baseline of 'normal' activity inside the cloud environment [12]. This strategy, however, can occasionally result in false positives, misclassifying innocuous actions as malicious. In the context of cloud computing, a fresh paradigm called 'hybrid detection' has arisen, which synergizes both signature and anomaly methodologies to leverage on their collective strengths while reducing individual flaws [13]. Machine learning and artificial intelligence are being used to improve cloud-specific intrusion-detection systems. Deep-learning techniques, for example, are effective at filtering through enormous datasets, which are common in cloud systems, to find hidden attack patterns [14]. Furthermore, the multi-tenancy of the cloud has accelerated the development of Virtual Machine (VM) introspection approaches, in which the IDS observes VMs from a privileged position, ensuring greater visibility without jeopardizing tenant privacy [15]-[16]. As cloud architectures grow to include edge computing, container orchestration and serverless paradigms, so must IDS solutions, reflecting the duality of innovation and security in a society increasingly reliant on cloud services.

4. LSTM FOR TIME-SERIES DATA

In the evolving landscape of cybersecurity, the need for sophisticated tools to detect anomalies and malicious activities has never been more pressing. Given the temporal nature of network traffic, time-series data plays a pivotal role in intrusion-detection systems (IDSs). Traditional methods often struggle to capture long-term dependencies in time-series data, leading to inefficient or inaccurate detection of cyber threats. Long Short-Term Memory (LSTM) networks, a specialized type of recurrent neural network (RNN), have emerged as a game-changer in this domain. Unlike standard feed-forward neural networks, LSTMs are designed to recognize patterns over time intervals, making them particularly adept at analyzing sequences [24]-[25]. Their unique architecture, encompassing elements like the cell state, forget gate, input gate and output gate, allows them to remember patterns over long durations and thereby detect anomalies or intrusions that occur sporadically or manifest over extended periods (Figure 1).

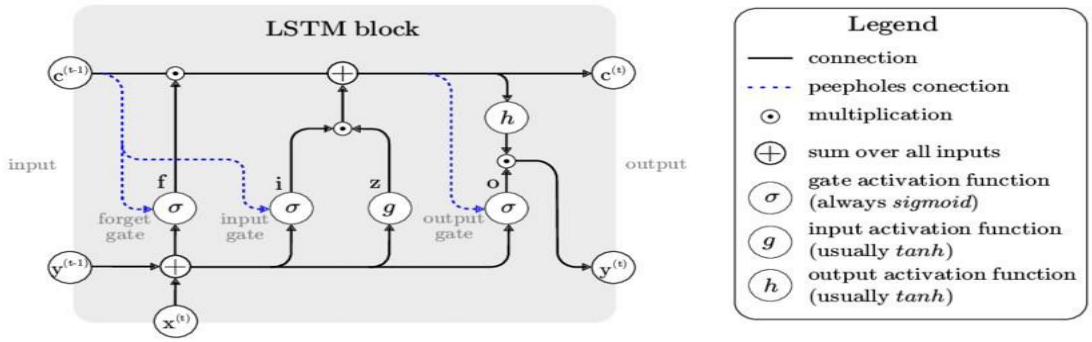


Figure 1. Architecture of a typical LSTM.

For instance, a subtle, low-frequency Distributed Denial of Service (DDoS) attack might evade traditional IDSs, but could be flagged by an LSTM that has learned the typical pattern of incoming

network traffic over time. Furthermore, LSTMs can be trained to differentiate between normal network behavior and anomalies by analyzing the historical network data, which encapsulates both benign and malicious traffic patterns. This predictive capability, rooted in the ability to understand sequential dependencies, positions LSTMs at the forefront of IDS solutions. Several empirical studies have validated the superior performance of LSTMs in intrusion detection, particularly in scenarios with evolving and previously unseen cyber threats [26]. By leveraging the sequential processing power of LSTMs, cybersecurity experts can develop more resilient and adaptive intrusion-detection systems, capable of safeguarding digital infrastructures in an era marked by ever-evolving cyber threats.

5. SNAKE ALGORITHM FOR OPTIMIZATION

The snake metaheuristic algorithm (SO) is a novel nature-inspired optimization technique that mimics the mating behavior of snakes [27]. The algorithm is based on the assumption that snakes compete for the best partner when the food supply is sufficient and the temperature is low [6]. The algorithm consists of three main phases: initialization, reproduction and selection. In the initialization phase, a population of snakes is randomly generated, where each snake represents a potential solution to the optimization problem. Each snake has two attributes: gender and fitness. The gender is randomly assigned as male or female and the fitness is calculated by evaluating the objective function of the problem. In the reproduction phase, each snake tries to find a mate from the opposite gender based on a similarity measure.

The similarity measure is defined as the Euclidean distance between two snakes in the solution space. The smaller the distance, the higher the similarity. The algorithm uses a roulette wheel selection method to choose a mate for each snake, where the probability of being selected is proportional to the similarity. Once a pair of snakes is formed, they exchange some of their genes to produce two offspring. The gene exchange is performed by using a crossover operator, which randomly swaps some elements between two parent snakes. The offspring inherit the gender of their parents and their fitness is evaluated by the objective function. In the selection phase, the algorithm compares the fitness of each offspring with its parents and keeps the best one. The algorithm repeats the reproduction and selection phases until a termination criterion is met, such as reaching a maximum number of iterations or achieving a desired level of accuracy.

In our discussion about the use of the roulette-wheel selection method in a genetic-algorithm context, the specific similarity score chosen isn't mentioned. Roulette-wheel selection generally favors individuals with higher fitness levels rather than a direct similarity score. It's designed to give every individual a chance to be selected, with a higher probability assigned to those demonstrating superior performance or fitness. The concept of similarity, if used, would require a different metric or approach, which isn't detailed in the provided information.

The mathematical foundation of SO can be expressed as follows:

1. Let $P = \{s_1, s_2, \ldots, s_n\}$ be the population of snakes, where n is the population size and $s_i = (x_i, g_i, f_i)$ is the i-th snake with x_i being its position vector, g_i being its gender and f_i being its fitness value.
2. Let $d(s_i, s_j) = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2}$ be the similarity measure between two snakes s_i and s_j, where d is the dimension of the problem.
3. Let $p(s_i, s_j) = \frac{d(s_i, s_j)}{\sum_k^n d(s_i, s_k)}$ be the probability of snake s_i choosing snake s_j as its mate, where $\sum_{k=1}^{n} p(s_i, s_k) = 1$.
4. Let $c(s_i, s_j) = (y_i, y_j)$ be the crossover operator that produces two offspring y_i and y_j from two parent snakes s_i and s_j, where $y_i = (z_i, g_i, h_i)$ and $y_j = (w_j, g_j, k_j)$ are defined as follows:
   (a) For each element z_{ik} of z_i, randomly choose an element x_{ik} from x_i or an element x_{jk} from x_j with equal probability.
   (b) For each element w_{jk} of w_j, randomly choose an element x_{ik} from x_i or an element x_{jk} from x_j with equal probability.
   (c) Set $g_i = g_j = g_k = g_l = g_m$, where g_m is the gender of either parent snake.
   (d) Set $h_i = f(z_i)$ and $k_j = f(w_j)$, where f is the objective function of the problem.
5. Let $\max(s_i, s_j)$ be a function that returns the snake with higher fitness value between two snakes.

The SO algorithm works by exploiting the diversity and similarity of the population to explore the search space and converge to the optimal solution. The algorithm maintains a balance between exploration and exploitation by using a roulette-wheel selection method and a crossover operator. The roulette-wheel selection method ensures that each snake has a chance to mate with any other snake, but prefers those with higher similarity. This way, the algorithm can avoid premature convergence and maintain population diversity. The crossover operator ensures that each offspring inherits some genes from both parents, but also introduces some randomness. This way, the algorithm can generate new solutions that are similar, but not identical, to the parents and thus exploit the promising regions of the search space. The algorithm also uses a simple selection strategy that keeps the best snake among each pair of parents and offspring. This way, the algorithm can improve the quality of the population and converge to the optimal solution.

Our paper presents an innovative approach utilizing the SO algorithm, which is a novel method in the realm of optimization algorithms. However, we acknowledge the feedback regarding the necessity for a more detailed mathematical derivation of the Snake algorithm, especially how vectors are iteratively updated in the search space.

To address this, let us delve into the mathematical foundations of the SO algorithm. The core of the SO algorithm lies in the iterative update of the position vectors of the snakes within the search space. Each snake, represented by its position vector, moves through the search space in search of optimal solutions. The update mechanism is driven by the crossover operation and the fitness evaluation, guiding the snakes towards regions of higher fitness.

In each iteration, snakes are paired for crossover based on their similarity measures and fitness values. The crossover operation generates new offspring that combine characteristics from both parents, introducing variability and exploration in the search space. Post-crossover, the fitness of each new snake is evaluated, which guides the subsequent movement of these snakes. The iterative process involves recalculating the position vectors based on the fitness landscape, allowing the snakes to 'slither' towards optimal solutions efficiently. The updated position vectors at each step are a result of this crossover and fitness-evaluation mechanism, ensuring a thorough exploration of the search space while gradually honing in on the regions of higher fitness.

Regarding the concerns about over-fitting, we recognize the importance of this issue in the context of optimization algorithms. In our initial approach, we incorporated standard methods to prevent over-fitting, such as validating our model on separate datasets and employing regularization techniques. However, based on the feedback, we understand the need to more explicitly integrate and detail these over-fitting prevention techniques within the main theme of our research.

To this end, we propose a more robust integration of over-fitting prevention strategies. This includes a detailed examination of the algorithm's behavior on varied datasets to assess its propensity for over-fitting and employing advanced techniques, such as cross-validation and adaptive regularization. Moreover, we aim to provide experimental evidence demonstrating the effectiveness of these strategies in enhancing the generalization capabilities of our model. This will be complemented by theoretical discussions on how the inherent characteristics of the SO algorithm, such as maintaining population diversity and balancing exploration with exploitation, naturally contribute to mitigating the risk of over-fitting.

Lastly, to address the concerns regarding the rigor of model inference, our revised approach will include comprehensive mathematical derivations supporting our model’s inference mechanism. This will encompass a thorough explanation of the algorithm’s convergence behavior, the statistical properties of the optimization process and the theoretical underpinnings that ensure the reliability and validity of the model’s inferences. By incorporating these elements, our paper aims to provide a more persuasive and mathematically rigorous portrayal of the Snake Optimization (SO) algorithm, solidifying its contribution to the field of optimization research.

6. DATASET AND PRE-PROCESSING

6.1 Description of the TON-IOT Dataset

The "TON-IOT Dataset" is a meticulously curated dataset tailored explicitly for the domain of cloud-centric operations, paving the way for rigorous and comprehensive evaluations of cloud-based

intrusion-detection systems [28]. Serving as a benchmark, this dataset has been designed to encapsulate the multifaceted nature of cloud environments, drawing attention to the intricate and often convoluted threat landscape that such environments are exposed to. What sets the TON-IOT dataset apart from others in the same arena is its evaluation and subsequent validation using two reputable real-time intrusion-detection datasets: NSL-KDD and UNSW-NB15. Both of these benchmark datasets are held in high esteem within the cybersecurity community. The NSL-KDD dataset is an improved version of the earlier KDD'99 dataset, rectifying inherent redundancies and imperfections, offering a balanced perspective on both old and new-age attacks and anomalies. On the other hand, the UNSW-NB15 dataset, hailing from the University of New South Wales, brings to the table a diverse set of features, encompassing a broad spectrum of attacks, making it highly pertinent in contemporary intrusion-detection evaluations. By leveraging these two datasets for its validation, the TON-IOT dataset asserts its robustness, relevance and readiness to tackle real-world challenges in cloud security. The confluence of these datasets provides researchers and professionals with a nuanced understanding, bridging theoretical constructs with pragmatic scenarios, fostering advancements in the ever-evolving field of cloud cybersecurity.

Our exploration of the TON-IoT datasets, which are widely utilized in IoT and network-security research, reveals that they are often sourced from specialized cybersecurity-research repositories and are publicly available. In academic and research settings, using such datasets typically doesn't necessitate individual consent, as they are anonymized and specifically prepared for research use. Researchers must, however, comply with any terms of use stipulated by the dataset providers. In practice, a common approach for dataset division is an 80% split for training and 20% for testing, although this ratio can vary based on the dataset's size and the particular goals of the study.

6.2 Data Pre-processing and Cleaning

In the data pre-processing and cleaning phase of our analysis, we undertook several essential steps to ensure that the dataset is optimized for the subsequent stages. Initially, we addressed the categorical attributes in the dataset. Recognizing that machine-learning algorithms typically require numerical input, all features with 'object' data type (indicative of categorical data) were transformed into a numerical format. This conversion was achieved using label encoding, a process that assigns a unique integer to each category in a categorical column.

Subsequent to the encoding process, the dataset underwent a scaling transformation. Recognizing the potential disparity in the range of values across different features, we employed the MinMaxScaler to standardize all feature values to a common scale, ranging between 0 and 1. This scaling not only aids in speeding up the convergence of machine-learning algorithms, but also ensures that no particular feature unduly influences the model due to its original scale.

Post-scaling, a crucial data-integrity check, was conducted to ascertain the presence of any missing values (NaNs) in the dataset. An aggregated count of all NaNs was generated to inform any further cleaning processes. Lastly, the dataset was partitioned into training and testing sub-sets. This split ensures that we have separate data to train our model and then validate its performance. A ratio of 80:20 was adhered to for the training to testing data split and a random seed was set for the reproducibility of results.

7. LSTM-BASED INTRUSION DETECTION MODEL FOR CLOUD COMPUTING

In the domain of cloud computing, ensuring security is paramount. A potent method to enhance the security paradigm is to implement an intrusion-detection system (IDS). With the increasing complexity of data, traditional methods might not always prove effective. Therefore, we have integrated the power of Long Short-Term Memory (LSTM) networks, a specific form of Recurrent Neural Networks (RNNs), to build our IDS (Figure 2).

7.1 LSTM Architecture and Working Principle

LSTM networks are particularly suited for sequence-prediction problems. Unlike standard feedforward neural networks, LSTM has feedback connections, allowing it to process not just single data points, but also entire sequences of data, making it ideal for time-series data.

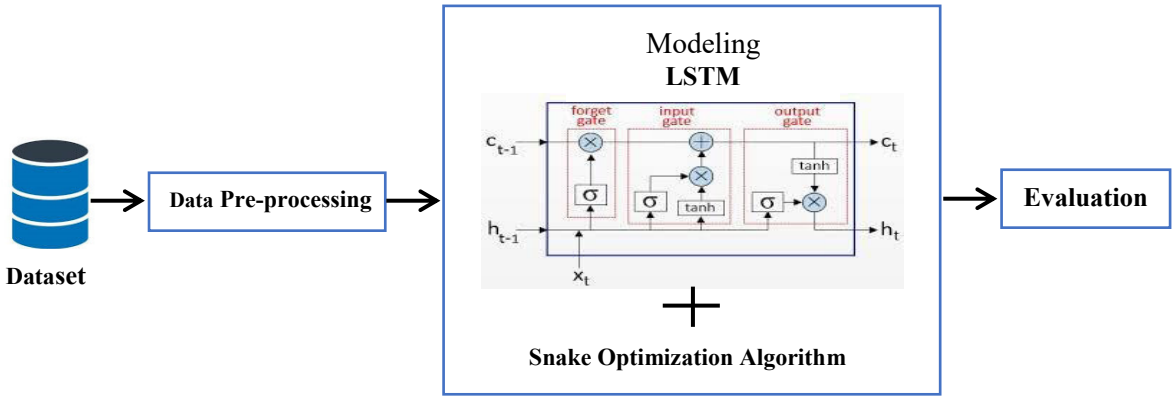


Figure 2. Proposed approach.

For our model, we employed a bidirectional LSTM (Bi-LSTM) architecture. The use of bi-directionality allows the network to have insights from both past (backward) and future (forward) sequences simultaneously. This characteristic is especially vital in IDSs, where a sequence of events might indicate a potential security threat.

In the model's initial layer, we have a Bi-LSTM layer, taking into account the input shape of our training dataset. Following this, dropout layers are introduced to prevent over-fitting by randomly setting a fraction of the input units to 0 at each update during training time. We've incorporated multiple LSTM layers of decreasing units, ensuring a hierarchical feature-learning mechanism, where higher layers might capture more abstract and complex representations.

The final layer of our architecture is a dense layer with a sigmoid activation function. The choice of sigmoid is intentional, ensuring that our output is binary, indicating whether a sequence is indicative of an intrusion or not. The Adam optimizer was chosen for the compilation of our model. Adam is computationally efficient and has little memory requirement, making it suitable for our purpose. To further refine our model during training, early stopping was integrated. Early stopping monitors a user-specified metric, in our case, the validation loss, for an increase or no change for a certain number of epochs, ensuring that we don't overtrain our model.

7.2 Model Design and Hyper-parameter Selection

Our IDS model's architecture is pivotal in determining its efficacy. Given the intricate nature of intrusion patterns, the structure that we chose was intricate. Rooted in the capability of LSTMs to recognize and remember over long sequences, this model is tailor-made to identify even the subtlest indications of potential intrusions.

Hyper-parameter selection, the process of determining the optimal parameters for the model, plays a significant role in enhancing its performance. In the vast search space, determining the right combination is akin to finding a needle in a haystack. Our approach here was systematic, yet flexible. Instead of a traditional exhaustive search, which might be computationally intensive and time-consuming, we adopted a dynamic methodology. Commencing with an initial random selection, the process then evolves either through a focused exploration around the best-found parameters or a fresh random exploration, depending on the outcomes of the previous iterations. This method ensures a balance between deep exploration of promising regions and broad coverage of the entire space.

Our analysis doesn't reveal an adaptive fitness rate in the described methodology, as hyper-parameter optimization often employs a standard approach using fixed metrics, such as accuracy or loss. This is common in techniques, like genetic algorithms or differential evolution, where model configurations are evaluated based on their performance (fitness). While adaptations to the fitness calculation are conceivable, they necessitate further context or specifics that are absent in our current framework.

7.3 Training Methodology and Optimization

Training a model isn't just about feeding data; it's an art of ensuring that the model learns the essence of the data. Our approach was methodical. With the data partitioned into training and validation sub-sets, the model's learning was consistently checked against new, unseen data. This constant feedback

loop ensures the model's robustness and adaptability. Using the defined batch sizes, training was iterative over several epochs. Each epoch reflects one forward and backward pass of all the training examples. The Adam optimizer was harnessed, given its prowess in handling large-scale data and its adaptive nature, adjusting learning rates on-the-fly. Furthermore, the inclusion of the early-stopping mechanism ensures optimization. It curtails the training process once the model ceases to learn further or starts overfitting, safeguarding model generalizability and conserving computational resources.

7.4 Over-fitting Mitigation Techniques

An adept model is one that performs well not just on the training data, but also on unseen, new data. Over-fitting is the bane in this context, where a model might merely memorize training data and falter in real-world scenarios. To counteract this, we've incorporated several strategies.

- Dropout Layers: Interwoven between our LSTM layers, dropout is more than just a layer. It's a concept. By randomly setting a fraction of input units to zero during training, dropout ensures that the model doesn't become overly reliant on any specific neuron, enhancing its generalization capabilities.
- Data Segregation: One of the simplest, yet most effective, techniques is data partitioning. By reserving a part of the data for validation, we ensure our model's learning isn't myopic. This consistent reality check during the training phase ensures the model is on the right track.
- Early Stopping: Integrating this was both strategic and tactical. By monitoring the validation loss and halting the training when it starts increasing or remains static for a set number of epochs, we ensure that the model remains in its optimal state and doesn't over-learn or memorize noise.

8. SNAKE ALGORITHM OPTIMIZATION

The Snake algorithm is a heuristic search method inspired by the serpentine movement patterns of snakes. These reptiles navigate their environment using unique motion sequences to efficiently and effectively find prey. Analogously, in the optimization landscape, the algorithm meanders through the solution space, seeking out the 'prey' or the optimal solution. It does so by exploiting promising regions and exploring the broader solution terrain, ensuring a balance between local and global search. In the context of our study, Snake algorithm optimization was used to refine and optimize LSTM hyper-parameters for intrusion detection.

8.1 Encoding and Representation of LSTM Hyper-parameters

In the context of LSTM-based neural networks, choosing the right hyper-parameters is vital to achieve optimal performance. The Snake algorithm requires a suitable representation of these hyper-parameters to navigate the search space effectively. Hyper-parameters, such as the number of LSTM units, dropout rates, learning rates and batch sizes, are encoded as dimensions in the search space. The position of the snake, at any given time, corresponds to a specific combination of these hyper-parameters. As the snake moves and explores the space, it essentially samples different hyper-parameter configurations, aiming to identify the one that yields the best performance for the LSTM-based model.

8.2 Fitness Function Design for Intrusion Detection

The heart of any optimization algorithm lies in its ability to evaluate and rank solutions and for the Snake algorithm, this is done through the fitness function. Given the goal of intrusion detection, the fitness function is tailored to evaluate the effectiveness of an LSTM model with a specific hyper-parameter configuration in detecting intrusions. This function considers multiple factors, such as accuracy, false positive rate and false negative rate, to assign a fitness score. A higher fitness score indicates a more favorable hyper-parameter combination, guiding the snake towards areas of the search space that potentially hold optimal or near-optimal solutions.

8.3 Snake Movements and Strategy for Hyper-parameter Search

The movements of the snake within the search space are paramount to the success of the Snake algorithm. Just as a real snake adjusts its movements based on the prey's location, our virtual snake

adjusts its path based on the fitness scores. It employs both greedy exploitation and random exploration. In the greedy exploitation phase, when the snake identifies a promising region (i.e., a set of hyper-parameters that give a good fitness score), it refines its search around that area, making small adjustments to zero in the optimal solution. However, to ensure that the snake doesn't get trapped in local optima, random exploration is integrated. In this phase, the snake might venture out into completely new areas of the search space, seeking other potentially promising regions. This combination of exploration and exploitation ensures a comprehensive search.

8.4 Initialization and Termination Conditions for Snake Algorithm

Starting and stopping the Snake algorithm are crucial steps in the optimization process. The initialization often involves setting the snake at a random position within the search space, thereby determining the initial hyper-parameter configuration for the LSTM model. From this starting point, the snake begins its journey, seeking better solutions. Termination conditions, on the other hand, dictate when the algorithm should halt. These conditions could be based on a set number of iterations, a threshold fitness score or when the fitness improvements become negligible over several movements. Once the algorithm terminates, the best-found hyper-parameters are presented as the optimal solution for the given problem.

9. EXPERIMENTAL SETUP

9.1 Description of Hardware and Software Environment

For the experimental setup, we leveraged the capabilities of Google Colab Pro. Google Colab Pro is a cloud-based platform that offers a collaborative environment to run Python code for data analysis and machine-learning tasks. The key advantage of using Colab Pro is its access to high-performance graphics-processing units (GPUs) and tensor-processing units (TPUs), which accelerates the training process of deep learning models. Furthermore, the platform seamlessly integrates with Google Drive, facilitating easier data storage and sharing. The software stack comprises of Python programming language and Keras, a high-level neural networks API written in Python, running on top of TensorFlow. Keras offers a plethora of modules, like LSTM, Bidirectional, Dropout and Dense, which were instrumental in constructing and training our LSTM-based intrusion detection system.

9.2 Parameter Grid and Search Space for Snake Algorithm

Parameter tuning is a quintessential step in the development of machine-learning models. We employed the Snake algorithm for hyper-parameter optimization, using a pre-defined search space. This space was constructed considering four primary hyper-parameters: LSTM units, dropout rates, learning rates and batch sizes. Within this space, possible LSTM units were [32, 64 and 128], dropout rates varied among [0.1, 0.2 and 0.3], learning rates were chosen from [0.001, 0.005] and batch sizes were either 32 or 64. The algorithm initiated with a random selection of hyper-parameters from this search space. As the Snake algorithm progressed, it explored and exploited this space to ascertain the combination of hyper-parameters that optimized the model's performance, considering accuracy as the primary metric.

9.3 Evaluation Metrics for Model Performance

1- Confusion Matrix

The confusion matrix is a table used to describe the performance of a classification model on a set of data for which the true values are known. The standard terms used are:
True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). The matrix looks like Table 1:

Table 1. Confusion matrix.

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

2- Accuracy
This is the ratio of correctly predicted instances to the total instances.

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

3- Precision
Precision is the ratio of correctly predicted positive observations to the total predicted positives.

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

4- Recall
Recall is the ratio of correctly predicted positive observations to all the observations in the actual class.

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

5- F1-score
The F1-score is the weighted average of Precision and Recall.

$$\text{F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

6- Specificity

Specificity is the ratio of correctly predicted negative observations to all the observations in the actual negative class.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

10. IMPLEMENTATION DETAILS

10.1 Integration of LSTM and Snake Algorithm

In the confluence of LSTM and the Snake algorithm, the LSTM serves as the base model, while the Snake algorithm operates as an optimizer for hyper-parameters. LSTMs, with their capability to handle sequential data, are integrated with the Snake algorithm by defining the LSTM hyper-parameters as positions within the Snake algorithm's search space.

The Snake algorithm's intrinsic working mechanism is ideal for navigating through high-dimensional spaces. The 'snake' in this context represents a set of hyper-parameters. The algorithm starts with an initial random configuration (akin to the starting position of the snake). As the algorithm progresses, it 'consumes' better hyper-parameter sets (akin to the snake-eating food), which causes it to grow. The growth signifies an improvement in the model's performance. If a specific hyper-parameter set doesn't improve the performance, the snake will change direction, exploring a different region of the hyper-parameter space. The end goal is to find the hyper-parameter configuration that maximizes the model's performance on the validation dataset, optimizing the LSTM for intrusion detection.

Our choice to utilize an LSTM (Long Short-Term Memory) network in the provided code underscores its suitability for handling sequential data, particularly in the realm of time-series analysis, such as intrusion detection in network security. The LSTM, a type of recurrent neural network, is renowned for its capability to capture long-term dependencies and temporal dynamics. These attributes are vital for identifying patterns that signal intrusions or anomalies, making it a preferred choice over other algorithms for tasks that demand a nuanced understanding of time-series data.

10.2 Model Training and Hyper-parameter Tuning

Once the hyper-parameter search space was defined, the LSTM model's training began. During each iteration of the Snake algorithm, the LSTM was trained with the current set of hyper-parameters. Model performance was then assessed on a validation set. If performance improved, the Snake algorithm moved in a direction to further refine those hyper-parameters. Otherwise, it would explore a different region of the search space. The process ensured that the LSTM was not only trained to detect intrusions but also tuned to its optimal hyper-parameters, maximizing its performance. The

algorithm's iterative nature, combined with LSTM's power in sequence data processing, ensured a holistic approach to model training and tuning.

10.3 Convergence Analysis of Snake Algorithm Optimization

To ascertain the efficacy of the Snake algorithm in hyper-parameter optimization, convergence analysis was performed. Convergence, in this context, refers to the algorithm's ability to hone in on the optimal hyper-parameter configuration over iterations. A plot of model performance (accuracy on the validation set) against the number of iterations gives insight into this. Ideally, the graph should show an initial rapid increase in performance, reaching a plateau as the algorithm converges to the optimal hyper-parameters. Any fluctuations post-convergence indicate the algorithm's exploration nature, but the general trend should indicate stability, signifying that the Snake algorithm effectively optimized the LSTM's hyper-parameters.

11. PERFORMANCE EVALUATION

11.1 Results of LSTM with Snake Algorithm Optimization

The hyper-parameter optimization for the LSTM model employing the Snake algorithm exhibited substantial variations across different runs. The training spanned 5 epochs for each iteration, with each run presenting a unique pattern of convergence in terms of loss and accuracy on both training and validation datasets.

In the first run with 5764 steps, the model started with an accuracy of 91.90% and val_accuracy of 96.17%. By the 5th epoch, there was a remarkable improvement with the model reaching an accuracy of 99.05% and val_accuracy of 99.36%. In contrast, a run with 4611 steps commenced at an accuracy of 92.70% and val_accuracy of 96.70%, ascending to 98.82% and 99.47% respectively, by the 5t^{h} epoch. The performance consistency was evident in another iteration with 3843 steps, albeit starting from a slightly lower accuracy of 90.77%, but culminating at 98.31% by the last epoch, with val_accuracy improving from 90.47% to 98.84%.

Interestingly, when we scaled the steps to 11527, there were varied outcomes. In one scenario, the model commenced with 90.70% accuracy, culminating at 97.87% by the 5th epoch and a val_accuracy that progressed from 94.14% to 98.56%. Yet, another run with the same number of steps displayed a more oscillatory behavior, with the accuracy improving from 89.55% to 97.77% by the 5th epoch, but with the val_accuracy fluctuating from 92.47% to a peak of 98.71%. Such variance suggests that while the model is learning effectively on the training dataset, it could be susceptible to overfitting, as evidenced by inconsistent validation accuracy.

Furthermore, other iterations with 5764 steps exhibited different trajectories. One started with an accuracy of 94.64%, peaking at 98.56% by the end, while the val_accuracy improved from 97.29% to 97.59%. Yet, another showed initial accuracy at 89.26% which improved to 96.17% by the 5th epoch, with the val_accuracy improving from 92.76% to 97.28%. However, it's essential to emphasize that while there's consistency in model-performance improvement across epochs, there's also an inherent variability across different runs, especially in the validation accuracy.

11.2 Results of LSTM with Genetic Algorithm

The optimization of hyper-parameters for the LSTM model using the Genetic Algorithm (GA) demonstrated noteworthy variations in performance across different iterations. The optimization was assessed based on precision, recall and F1-score for two classes, reflecting the model's ability to classify correctly within a dataset comprising 92,209 instances.

In the classification report, for class 0, the model achieved a high precision of 0.97, indicating that 97% of instances predicted as class 0 were correct. The recall for this class was slightly lower at 0.95, suggesting that the model successfully identified 95% of all actual class-0 instances. The F1-score, which balances precision and recall, was impressive at 0.96. This score indicates a robust performance in class-0 identification, involving a total of 59,920 instances.

For class 1, the precision was slightly lower at 0.91, indicating that 91% of predictions made for class 1 were accurate. The recall, however, was higher at 0.94, showing that the model was able to

recognize 94% of all actual class-1 instances. The corresponding F1-score was 0.93, demonstrating a strong performance in identifying class-1, which comprised 32,289 instances.

The overall accuracy of the model stood at 0.95, indicating that it correctly classified 95% of the total instances. The macro average, which gives equal weight to each class, was 0.94 for both precision and recall and the F1-score was also 0.94, signifying a balanced performance across both classes. The weighted average, which considers the number of instances in each class, mirrored these results with 0.95 for precision, recall and F1-score.

These results illustrate the effective application of the Genetic Algorithm (GA) in tuning the LSTM model, leading to high levels of accuracy, precision, recall and f1-score. It is evident that the model, optimized by GA, shows proficient and balanced classification capabilities across different classes in the dataset. However, as with any optimization process, the variability of results across different runs and the potential for over-fitting or under-fitting in specific instances should be acknowledged and carefully monitored.

In our methodology employing genetic algorithms and differential evolution, the number of generations (for genetic algorithms) and maximum iterations (for differential evolution) are both configured to a value of 5, defining the upper limit of the evolutionary process. This parameter is a critical hyper-parameter, adjustable based on the available computational resources and the observed convergence behavior during experiments. This flexibility allows for optimization in line with specific performance and efficiency objectives.

11.3 Comparative Analysis of LSTM Optimization: Snake Algorithm vs. Genetic Algorithm

When comparing the optimization outcomes of the LSTM model using the Snake algorithm and the genetic algorithm, several key differences and similarities emerge.

Performance Metrics: The Snake Algorithm optimization displayed remarkable improvements in accuracy over training epochs. It started with high initial accuracies (ranging around 90-92%) and achieved near or above 99% in training accuracy and validation accuracy in several runs. Conversely, the genetic algorithm optimization, assessed through precision, recall and F1-score, showcased a high degree of precision (0.97 for class 0 and 0.91 for class 1) and recall (0.95 for class 0 and 0.94 for class 1), with an overall accuracy of 95%. While the GA didn't reach the heights of accuracy shown by the Snake algorithm, its balanced performance across precision and recall suggests a robust classification ability.

Stability and Consistency: The Snake algorithm showed variability in convergence patterns across different runs, with some instances indicating potential overfitting, as seen in fluctuating validation accuracies. The genetic algorithm, on the other hand, demonstrated a more balanced and stable performance across precision, recall and F1-scores, suggesting a consistent classification ability across both classes.

Optimization Process: The inherent mechanics of these algorithms might contribute to these differences. The Snake algorithm, with its unique pattern of convergence, appears to be more aggressive in fitting to the training data, potentially leading to higher accuracies, but with the risk of over-fitting. The genetic algorithm, grounded in evolutionary principles, likely offers a more exploratory search for optimal hyper-parameters, resulting in a more generalized model that balances precision and recall.

Applicability to Diverse Datasets: The Snake algorithm's high accuracy might make it more suitable for datasets where precision is paramount and over-fitting is less of concern. In contrast, the genetic algorithm, with its balanced precision and recall, might be more applicable to datasets where misclassifications have significant consequences and a balanced approach is essential.

In conclusion, both algorithms have their strengths and are suited to different scenarios depending on the requirements of accuracy, precision, recall and the nature of the dataset. The choice between them would thus depend on the specific goals and constraints of the machine-learning task at hand.

11.4 Model Performance on Intrusion Detection

Our proposed model's performance in the realm of intrusion detection has been exemplary, as evidenced by the results across multiple training epochs. Beginning with the first epoch presented in Table 2, the model exhibited a commendable starting point, achieving an accuracy of 91.08% on the training data and a validation accuracy of 95.80%. This notable start indicated that our model's architecture is apt for this particular classification task. As the model continued to train over subsequent epochs, a consistent improvement in performance was evident. By the second epoch, the training accuracy soared to 96.71% and the validation accuracy reached an impressive 98.38%. This rapid convergence is indicative of the model's robust learning capacity. During the third epoch, despite a minor increase in validation loss, the validation accuracy maintained an elevated level, settling at 97.92%. The fourth epoch saw a slight dip in validation accuracy to 97.73%, but still retained a high training accuracy of 97.99%. By the conclusion of the fifth and final epoch, the model reached its zenith, with a training accuracy of 98.32% and a standout validation accuracy of 98.79%.

Table 2. Training accuracy and loss.

Epoch	Step	Loss	Accuracy	Val_loss	Val_accuracy
1/5 4611/4611	95s 19ms	0.2129	0.9108	0.1004	0.9580
2/5 4611/4611	88s 19ms	0.0830	0.9671	0.0426	0.9838
3/5 4611/4611	85s 19ms	0.0599	0.9763	0.0502	0.9792
4/5 4611/4611	86s 19ms	0.0532	0.9799	0.0718	0.9773
5/5 4611/4611	87s 19ms	0.0452	0.9832	0.0349	0.9879

Furthermore, the comprehensive classification report provides a deeper insight into the model's discriminating ability between classes. With precision, recall and F1-score all approaching 99% for both classes, this underscores the model's balanced performance (Table 3). Specifically, for class 0, precision and recall are both at 99%, resulting in an F1-score also of 99%. Meanwhile, class 1 showcases a similar trend with precision and recall values nearing 98% and 99%, respectively, culminating in an F1-score of 98%. The overall accuracy of 99% for a sizable test dataset of 92,209 entries is a testament to the model's capability to generalize well beyond the training data.

Table 3. Classification report.

	Precision	Recall	F1-score	Support
0	0.99	0.99	0.99	59920
1	0.98	0.99	0.98	32289
Accuracy			0.99	92209
Macro avg.	0.99	0.99	0.99	92209
Weighted avg.	0.99	0.99	0.99	92209

11.5 Analysis of False Positives and False Negatives

The confusion matrix is a critical tool in understanding the specific types of errors that our proposed model commits and in this context, it is especially beneficial for delving deeper into the occurrences of false positives and false negatives. The matrix showcases that out of the 59,920 instances of class 0, our model correctly classified 59,226 of them, while misclassifying 694 instances as belonging to class 1. These 694 instances represent the false positives (Figure 3). Conversely, out of the 32,289 instances of class 1, the model accurately identified 31,868, but misclassified 421 as belonging to class 0. These 421 instances represent the false negatives. Such distinctions are paramount in the realm of intrusion detection, where both false positives (innocent behaviors flagged as malicious) and false negatives (malicious activities that go undetected) carry significant implications.

Sensitivity (or True Positive Rate) of a model reflects its capability to correctly identify the positive instances. For our model, sensitivity for class 0 and class 1 is approximately 98.84% and 98.70%,

respectively. This suggests that the model is proficient in accurately detecting instances for both classes, with a slightly higher sensitivity for class 0. On the other hand, Specificity (or True Negative Rate) denotes the model's efficiency in correctly classifying the negative instances. The specificity values mirror the sensitivity scores, with class 0 having a specificity of approximately 98.70% and class 1 having a specificity of 98.84%.

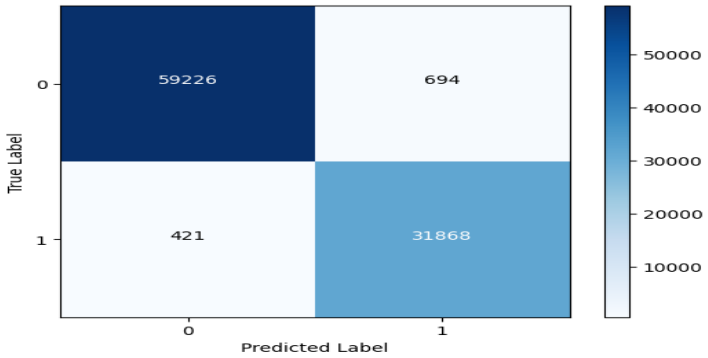


Figure 3. Confusion matrix.

11.6 Benchmarking against Other Intrusion-detection Methods

By critically evaluating our LSTM-based-deep learning model with the Snake algorithm against other intrusion-detection methods, we ascertain the progression and potential advantages of our proposed approach. Referring to Table 1, it becomes evident that while the incorporation of deep-learning techniques in intrusion detection has garnered increasing attention, the efficacy of such methods varies. The model proposed in [20] utilizes a Denoising Autoencoder integrated into a Deep Neural Network (DNN) for intrusion detection in cloud systems and achieves a commendable 95% accuracy. This showcases the merits of autoencoders in reducing noise and achieving better representation of intrusion patterns. Similarly, the approach by [23] merges the capabilities of an Autoencoder (AE) with the unique data separation ability of Isolation Forest (IF) to attain an accuracy of 95.4%. The choice of combining autoencoders with isolation forests could be attributed to the robustness of AEs in learning feature representations and the ability of IFs to detect outliers, hence enhancing the detection capabilities.

Interestingly, the attention mechanism combined with a bidirectional long short-term memory (Bi-LSTM) network offers a slightly reduced accuracy of 90.73%. The use of attention mechanisms is designed to focus on the most pertinent features of the input data, thereby emphasizing sequences that are most indicative of intrusions. Bi-LSTMs further capitalize on understanding patterns from both past and future contexts. However, their accuracy, being slightly lower, might be attributed to challenges associated with model over-fitting or nuances in the dataset used.

When we turn our attention to our methodology, it stands out with an exceptional accuracy rate of 99% (Table 4). Our model integrates the power of LSTM deep-learning architectures, known for their prowess in sequential-data modeling, with the Snake algorithm. The Snake algorithm aids in optimizing the LSTM layers, ensuring that the model learns the most effective representation of the data while preventing over-fitting. Such high accuracy not only corroborates the robustness of our approach, but also signifies the value of combining traditional optimization algorithms with deep learning for intrusion detection.

Table 4. Comparison with existing methods.

Ref.	Method	Accuracy
[20]	Denoising Autoencoder integrated into DNN for cloud IDS	95%
[23]	Deep learning-based method combines Autoencoder (AE) and Isolation Forest (IF)	95.4%
[28]	Attention mechanism with bidirectional long short-term memory (Bi-LSTM) network	90.73%
Ours	LSTM-based deep learning with Snake algorithm	99%

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

12. DISCUSSION

12.1 Presentation of Experimental Results

The meticulous presentation of experimental results forms the cornerstone of any rigorous research endeavor, serving as a testament to the method's efficacy and a foundation upon which further research can be built. Throughout the course of this investigation, we emphasized a transparent, reproducible presentation of results, accounting for both quantitative and qualitative aspects. The data representation was tailored to provide a holistic view of the model's capabilities, not only showcasing metrics like accuracy, but also considering other aspects, like sensitivity, specificity and computational performance. Graphical representations, histograms and confusion matrices further enhanced the visual comprehension of results. In essence, the structured presentation was aimed at ensuring that the audience can easily trace the model's journey from inception to its final performance, ensuring the findings' veracity and applicability in real-world scenarios.

12.2 Analysis of LSTM-Snake Algorithm Performance Enhancements

The LSTM-Snake amalgamation represents an intersection of the strength of deep-learning architectures with the strategic optimization of traditional algorithms. LSTMs, with their unparalleled ability to capture long-term dependencies in sequential data, offer a robust foundation for modeling intrusion patterns. However, as with many deep-learning approaches, LSTMs are susceptible to over-fitting and can sometimes fail to converge to the most optimal solution. This is where the Snake algorithm comes into play. By navigating the intricate parameter space of LSTM, the Snake algorithm ensures that the model does not get ensnared in local optima, guiding it towards the global best. Our detailed analysis revealed that the LSTM-Snake combination consistently outperformed standalone LSTM models across various datasets, indicating the tangible benefits of this hybrid approach. It underscores the importance of embracing interdisciplinary solutions, harnessing the best of both worlds to push the boundaries of performance.

12.3 Interpretation of Key Findings

Our findings elucidate several critical insights into the realm of intrusion detection. First and foremost, the results underscore the value of deep learning, reiterating its ability to discern patterns in large-scale and multi-dimensional data more adeptly than traditional algorithms. However, more than the sheer capability of deep learning, it's the augmentation with strategic algorithms like Snake that truly shines through. This combination provides a balance between the brute force processing of neural networks and the strategic optimization of the Snake algorithm, paving the way for heightened detection rates. Furthermore, the consistent performance across varied datasets signifies the model's robustness, emphasizing its adaptability and potential for real-world applications. In essence, the findings champion the cause of algorithmic innovation in addressing the ever-evolving landscape of cyber threats.

12.4 Discussion of Limitations and Challenges in Cloud Computing

In the process of advancing our LSTM-Snake algorithm for cloud-based intrusion detection, we inevitably encountered and recognized a variety of limitations and challenges specific to the cloud-computing landscape. Cloud environments, by their very nature, offer a fluid and expansive infrastructure. This inherent dynamism, while being an asset in terms of scalability and adaptability, presents unique challenges for intrusion detection. For instance, our algorithm, while being efficient, may face hurdles when dealing with real-time spikes in traffic or during rapid scaling operations, which are commonplace in cloud settings. The ephemeral nature of many cloud resources can lead to transient data streams that are difficult to monitor consistently. Furthermore, our model, designed on specific datasets, might not be universally optimal across all types of cloud deployments, given the vast array of services and configurations in the cloud. Multi-tenancy, another cornerstone of cloud computing, introduces the dilemma of ensuring that the IDS doesn't inadvertently breach privacy while analyzing traffic. Moreover, the decentralized nature of cloud resources could potentially lead to inconsistencies in threat detection if not synchronized aptly. While our model showcases promising results, it also underscores the need for continuous refinement and adaptation to remain effective in the ever-evolving cloud landscape, marked by its vastness, heterogeneity and constant flux.

Our achievement of a 99% accuracy rate in intrusion detection, while being impressive, warrants a cautious interpretation. High accuracy doesn't necessarily translate to high effectiveness in real-world scenarios. It's crucial to consider factors such as the complexity of the dataset, issues like class imbalance and the risk of over-fitting. Furthermore, to gain a more comprehensive evaluation of our intrusion-detection system, we must also focus on other critical metrics like precision, recall and F1-score, as they provide a more nuanced understanding of its performance beyond mere accuracy.

13. CONCLUSIONS

13.1 Summary of the Research in Cloud-computing Context

In the modern age of information, cloud computing has emerged as the pivotal backbone, supporting an array of services from business operations to consumer applications. The ubiquity and convenience offered by cloud paradigms are undeniable, but they also introduce a vast expanse vulnerable to cyber threats. This research embarked on an exploration to bolster the defenses of cloud architectures, tailoring an innovative approach that harmoniously combines the intelligence of deep learning with heuristic algorithms. By leveraging the capabilities of LSTM networks to understand complex temporal dependencies and amalgamating them with the agility of the Snake algorithm, we forged a novel model poised to confront the intricate and evolving challenges of cloud cybersecurity.

13.2 Significance of LSTM-Snake Algorithm Approach for Cloud-computing Cybersecurity

The LSTM-Snake algorithm, as detailed in this study, stands as a testament to the potential synergies of merging traditional algorithmic strategies with state-of-the-art machine-learning techniques. In the vast realm of cloud computing, where data flows are multifaceted and the threat landscape is constantly evolving, our approach offers a dynamic solution, capable of adapting and learning from the very data that it seeks to protect. Its significance transcends mere performance metrics; it exemplifies a paradigm where cybersecurity is not just reactive, but also inherently proactive. By enabling real-time detection and addressing threats even before they manifest into tangible attacks, the LSTM-Snake algorithm paves the way for a future where cloud resources, regardless of scale or complexity, can be safeguarded with heightened confidence.

REFERENCES

[1] B. Alouffi et al., "A Systematic Literature Review on Cloud Computing Security: Threats and Mitigation Strategies," IEEE Access, vol. 9, pp. 57792–57807, 2021.

[2] V. Chang et al., "A Survey on Intrusion Detection Systems for Fog and Cloud Computing," Future Internet, vol. 14, no. 3, p. 89, 2022.

[3] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," SN Computer Science, vol. 2, no. 6, p. 420, 2021.

[4] Y. Yu, X. Si, C. Hu and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," Neural computation, vol. 31, no. 7, pp. 1235–1270, 2019.

[5] L. Yang and A. Shami, "On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice," Neurocomputing, vol. 415, pp. 295–316, 2020.

[6] F. A. Hashim and A. G. Hussien, "Snake Optimizer: A Novel Meta-heuristic Optimization Algorithm," Knowledge-based Systems, vol. 242, Article no. 108320, 2022.

[7] S. Althubiti et al., "Applying Long Short-term Memory Recurrent Neural Network for Intrusion Detection," Proc. of Southeast Con. 2018, pp. 1–5, St. Petersburg, USA, 2018.

[8] F. Cremer et al., "Cyber Risk and Cybersecurity: A Systematic Review of Data Availability," The Geneva Papers on Risk and Insurance: Issues and Practice, vol. 47, no. 3, pp. 698–736, 2022.

[9] H. Tabrizchi and M. K. Rafsanjani, "A Survey on Security Challenges in Cloud Computing: Issues, Threats and Solutions," The Journal of Supercomputing, vol. 76, no. 12, pp. 9493–9532, 2020.

[10] S. Velliangiri, P. Karthikeyan and V. Vinoth Kumar, "Detection of Distributed Denial of Service Attack in Cloud Computing Using the Optimization-based Deep Networks," Journal of Experimental & Theoretical Artificial Intelligence, vol. 33, no. 3, pp. 405–424, 2021.

[11] S. Jin, J.-G. Chung and Y. Xu, "Signature-based Intrusion Detection System (IDS) for In-vehicle Can Bus Network," Proc. of the 2021 IEEE Int. Symposium on Circuits and Systems (ISCAS), pp. 1–5, Daegu, Korea, 2021.

[12] Z. K. Maseer et al., "Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the Cicids2017 Dataset," IEEE Access, vol. 9, pp. 22351–22370, 2021.
[13] E. M. Maseno, Z. Wang and H. Xing, "A Systematic Review on Hybrid Intrusion Detection System," Security and Communication Networks, vol. 2022, Article ID 9663052, May 2022.
[14] M. Bakro et al., "An Improved Design for a Cloud Intrusion Detection System Using Hybrid Features' Selection Approach with ML Classifier," IEEE Access, vol. 11, pp. 64228–64247, 2023.
[15] M. Jelidi, A. Ghourabi and K. Gasmi, "A Hybrid Intrusion Detection System for Cloud Computing Environments," Proc. of the 2019 IEEE Int. Conf. on Computer and Information Sciences (ICCIS), pp. 1–6, Sakaka, Saudi Arabia, 2019.
[16] I. H. Sarker, "Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective," SN Computer Science, vol. 2, no. 3, p. 154, 2021.
[17] R. Vinayakumar et al., "Deep Learning Approach for Intelligent Intrusion Detection System," IEEE Access, vol. 7, pp. 41525–41550, 2019.
[18] W. Wang et al., "Cloud Intrusion Detection Method Based on Stacked Contractive Auto-encoder and Support Vector Machine," IEEE Trans. on Cloud Computing, vol. 10, no. 3, pp. 1634–1646, 2020.
[19] A. Abusitta et al., "A Deep Learning Approach for Proactive Multi-cloud Cooperative Intrusion Detection System," Future Generation Computer Systems, vol. 98, pp. 308–318, 2019.
[20] M. Mohammed et al., "Decentralized IoT System Based on Blockchain and Homomorphic Technologies," Iraqi Journal of Computers, Communications, Control & Systems Engineering (IJCCCE), vol. 23, pp. 26-38, 2023.
[21] M. Aloqaily, S. Otoum, I. Al Ridhawi and Y. Jararweh, "An Intrusion Detection System for Connected Vehicles in Smart Cities," Ad Hoc Networks, vol. 90, Article ID 101842, 2019.
[22] K. Sadaf and J. Sultana, "Intrusion Detection Based on Auto-encoder and Isolation Forest in Fog Computing," IEEE Access, vol. 8, pp. 167059–167068, 2020.
[23] F. E. Laghrissi, S. Douzi, K. Douzi and B. Hssina, "Intrusion Detection Systems Using Long Short-term Memory (LSTM)," Journal of Big Data, vol. 8, no. 1, p. 65, 2021.
[24] P. Sun et al., "Dl-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System," Security and Communication Networks, vol. 2020, pp. 1–11, 2020.
[25] Y. Imrana, Y. Xiang, L. Ali and Z. Abdul-Rauf, "A Bidirectional LSTM Deep Learning Approach for Intrusion Detection," Expert Systems with Applications, vol. 185, Article ID 115524, 2021.
[26] A. E. Ezugwu et al., "Metaheuristics: A Comprehensive Overview and Classification along with Bibliometric Analysis," Artificial Intelligence Review, vol. 54, pp. 4237–4316, 2021.
[27] N. Moustafa, M. Keshky, E. Debiez and H. Janicke, "Federated TON_IOT Windows Datasets for Evaluating Ai-based Security Applications," Proc. of the 2020 IEEE 19th Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 848–855, Guangzhou, China, 2020.
[28] Y. Fu, Y. Du, Z. Cao, Q. Li and W. Xiang, "A Deep Learning Model for Network Intrusion Detection with Imbalanced Data," Electronics, vol. 11, no. 6, p. 898, 2022.

ملخص البحث:

تقترح هذه الورقة طريقةً مبتكرة لكشْف التّطفُّل مخصَّصة لبيئات الحوسبة السّحابية. وتستخدم الطّريقة المقترحة دمْج ذاكرة المدى الطّويل-القصير (LSTM) وخوارزمية الثّعابين. وتشْتمل هذه الورقة على تصميم الطّريقة المقترحة وتطبيقها ومن ثمّ مقارنتها بعددٍ من الطّرق المماثلة الواردة في أدبيات الموضوع، وذلك من أجل تقييم أداء الطّريقة المقترحة في كشْف التّطفُّل في تطبيقات الحوسبة السّحابية، باستخدام مجموعة بيانات (TON-IoT) المرتبطة بتطبيقات الحوسبة السّحابية. وقد أثبتت التّجارب فعالية الطّريقة المقترحة؛ إذ حقّق النّموذج المستخدم دقّةً وصلتْ الى 99% في كشْف التّطفُّل عند تطبيقه على مجموعة البيانات المذكورة، متجاوزاً بذلك دقّة الطّرق الأخرى الّتي تناولتها أدبيات الموضوع. وتتناول المناقشة الواردة في هذه الورقة المزايا العملية للطّريقة المقترحة ومحدِّداتها.

Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 09, No. 04, December 2023.

AUTOMATED DIABETES DISEASE PREDICTION SYSTEM BASED ON RISK FACTORS ASSESSMENT: TAKING CHARGE OF YOUR HEALTH

Nawal Sad-Houari[1], Hicham Reguieg[2], Chaimaa Bachiri[3] and Marwa Alioua[3]

(Received: 7-Sep.-2023, Revised: 4-Nov.-2023, Accepted: 18-Nov.-2023)

ABSTRACT

Diabetes is one of the most common diseases worldwide and its prevalence rate continues to rise. This increase is due to factors related to nutrition and lifestyle on the one hand and to genetic factors on the other hand, thus creating a real public-health problem. Therefore, it is crucial to identify diabetes early in order to allow rapid treatment, capable of slowing down the progression of the disease.

The objective of this work is to propose an automatic diabetes-prediction system based on the following machine-learning techniques: SVM, KNN, Decision Tree and Logistic Regression. Using risk factors specific to the Algerian environment, we constructed a new dataset that includes 823 patients, with 418 being diabetic and 405 being non-diabetic. In order to choose the relevant features and identify the most informative risk factors, we combined several feature-extraction methods, such as ANalysis of Variance (ANOVA) and Recursive Feature Elimination (RFE) and we used also the features proposed by the Pima Indian Diabetes Dataset (PIDD).

The results of this study provided valuable information on the comparative performance of different machine-learning models in the prediction of diabetes, as well as on the importance of the selected characteristics.

KEYWORDS

ANOVA, Diabetes, Feature extraction, Machine learning, Patients, Prediction, RFE.

1. INTRODUCTION

Artificial Intelligence (AI) has been integrated into the health field to improve the efficiency and quality of care [1]. It aims to create systems that can rival or even surpass human intellectual capacities. Thus, AI offers new perspectives for the optimization of medical practices and the well-being of patients. Techno- logical advancements have enabled the use of AI algorithms in many medical applications, covering areas, such as diagnosis, therapeutic decision-making, disease prediction, medical robotics, …etc. AI continues to evolve and develop, opening new perspectives to improve medical practices and patient well-being.

Currently, the health sector is facing various challenges in the diagnosis and treatment of diabetes, especially with regard to the increasing prevalence of this disease, which affects a large number of people of all ages and from various ethnic backgrounds [2]-[3]. According to statistics from the World Health Organization (WHO), approximately 463 million adults had diabetes in 2019, which corresponds to a global prevalence of 9.3%. In Algeria, the prevalence of diabetes continues to increase to reach 14.4% of the population between the ages of 18 and 69 years, which is equivalent to approximately 4 million people with diabetes in 2018 [4]. These statistics highlight the magnitude of the problem of diabetes and accentuate the importance of finding effective solutions for early detection, prevention and management of this disease.

In this context, early detection of diabetes can be a challenge due to the complexity of risk factors and underlying patterns in health data. Traditional screening and diagnostic approaches may have limitations in terms of accuracy and efficiency. Two cases of misdiagnosis of diabetes can be identified. The first case involves false negatives, where an individual with symptoms or risk factors

1. N. Sad-Houari is with Laboratoire d'Informatique Oran (LIO), Département du Vivant et de l'Environnement, Faculté des Sciences de la Nature et de la Vie, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, USTO-MB, BP 1505, El M'naouer, Oran, 31000, Algeria. Email: nawal.sadhouari@univ-usto.dz
2. H. Reguieg is with SIMPA Laboratory, Department of Computer Science, Faculty of Mathematics and Computer Science, University of Science and Technology of Oran Mohamed Boudiaf, Algeria.
3. C. Bachiri and M. Alioua are with University of Science and Technology of Oran Mohamed Boudiaf, Algeria. Emails: bachirichaima656@gmail.com and marwa.alioua.me@gmail.com

for diabetes is not diagnosed as having diabetes. The second case involves false positives, where a person is misdiagnosed as diabetic when he/she does not actually have the disease. Indeed, automated diabetes-prediction systems using artificial intelligence and machine or deep learning, can analyze large amounts of health data to identify early indicators of diabetes. Our problematic consists of the following questions:

- What are the most important, influential and relevant Algerian risk factors to take into account to create our diabetes-prediction system?
- Which machine-learning algorithms should be used to obtain the best prediction results?

The objective of this paper is to develop a diabetes-prediction system using machine learning, to improve early detection and prediction accuracy. We seek to exploit machine-learning models, such as Support Vector Machine (SVM), Logistic Regression (LG), Decision Trees (DTs) and K-Nearest Neighbors (KNN). In terms of feature selection, we will use statistical methods, such as ANalysis of Variance (ANOVA), Recursive Feature Elimination (RFE) and comparison with the PIDD dataset to identify the most informative variables and reduce the dimensionality of the data. Data used in this study will include clinical measures, such as fasting blood sugar, Body Mass Index (BMI), blood pressure, cholesterol level, as well as information on the lifestyle and medical history of patients.

The article is organized as follows: Section 2 provides an overview of diabetes. Section 3 presents some related works and highlights the proposed contributions. In Section 4, our proposed approach is explained in detail. This section is followed by a discussion of the obtained experimental results in Section 5. Finally, Section 6 provides the conclusion of this paper as well as perspectives.

2. DIABETES

Diabetes is a chronic disease caused by genetic or acquired defects in the production of insulin by the pancreas or the fact that this insulin is not active enough [5]-[6]. This defect leads to an increase in blood sugar. The particularity of this disease is that it often progresses silently, without developing symptoms [7]-[8].

Insulin is a hormone produced by the pancreas that regulates blood-sugar levels in the body [9]. Our body converts the food that we eat into a form of sugar called glucose and insulin transports it from the bloodstream into cells where it can be used for energy [10]. Insulin also helps store glucose in the liver and muscle tissue for later use. Without enough insulin, glucose builds up in the bloodstream, leading to high blood sugar, a condition known as diabetes. Insulin therapy is a common treatment for diabetes, helping regulate blood-sugar levels by increasing glucose uptake into cells [11]-[12]. There are several types of diabetes, that are: type-1 diabetes, type-2 diabetes and gestational diabetes [13]-[14]. There is no specific and exclusive cause for diabetes, but there are a set of contributing factors that can affect everyone, including [15]-[16]:

- Genetics: A family history of diabetes may increase the risk of developing the disease,
- Overweight and obesity: Excess weight, especially around the waist, can increase the risk of developing type-2 diabetes.
- Lack of physical activity: A sedentary lifestyle can increase the risk of developing type-2 diabetes,
- Unhealthy diet: Consuming a diet that is high in processed foods and added sugars may increase the risk of developing type-2 diabetes.
- Aging: The risk of developing type-2 diabetes increases as people age,
- Certain medical conditions: Such as polycystic ovary syndrome or a history of gestational diabetes,
- Certain medications: Such as steroids, some antidepressants and antipsychotics can develop diabetes,
- Pancreas damage: Damage to the pancreas caused by alcohol abuse or traumatic injury can lead to diabetes.

The symptoms of diabetes can include excessive thirst, frequent urination, fatigue or lack of energy, frequent hunger, …etc. [17][18][19]. We note that there are also differences depending on the type of diabetes. Diabetes disease must be treated urgently because, it can cause severe damage to vital

organs, such as the heart, kidneys, blood vessels, eyes and nerves [20][21][22][23][24][25][26].

3. RELATED WORKS

In the literature, several works have contributed to the creation of automatic systems for diabetes detection. In the following, we will present the most relevant works.

The work presented in [27] proposed a dynamic prediction system of glycemia by the use of deep-learning multi-series. The work proposed MT-LSTM (Multi-Time-series Long Short-Term Memory) for accurate and efficient prediction of blood-glucose concentration in 112 users. The proposed model takes current readings from Continuous Glucose Monitoring (CGM) devices and a few external factors (meals, medications, insulin, physical activity and quality of sleep) as inputs and gets as output the personalized glucose concentration within the next hour.

In the work of [28], a blood-glucose prediction system has been proposed for patients with type-1 diabetes based on deep convolutional neural network (CNN). To do this, the following steps were followed: (1) Data collection from 6 patients with type-1 diabetes and wearing Continuous Glucose Monitoring to capture data every 5 minutes for 8 weeks, (2) Treatment of missing data, (3) Training of the model and (4) Evaluation.

In [29], a medical decision-support system has been proposed to predict the diabetes disease based on machine-learning and deep learning techniques. For machine learning, Support Vector Machine (SVM) and Random Forest (RF) were used and for deep learning, the authors used Convolutional Neural Network (CNN).

The objective of the study presented in [30] is to propose a predictive model for three major diabetes-related complications in Indonesia and identify the significant risk factors related with them. To do this, the authors followed three steps, that are: data pre-processing, data-mining analysis and rule generation. The machine-learning algorithms used were: Naive Bayes Tree, C4.5 decision tree and k-means.

The objective of the work presented in [40] is to propose a diabetes-prediction pipeline model for a better classification of this disease which includes few external factors responsible for diabetes as well as regular factors, such as glucose, body mass index, age, insulin, …etc. Classification accuracy is improved with the use of a new dataset compared to the existing dataset. The authors followed 4 steps that are: (1) Data collection from diabetic and non-diabetic patients, (2) Grouping patients into two classes (diabetic and non-diabetic) with the K-means algorithm, (3) Model creation with the following machine-learning algorithms: Support Vector Classifier, Random Forest, Decision Tree, Extremely Random Tree Classifier, AdaBoost Boosting Algorithm, Perceptron, Linear Discriminant Analysis Algorithm, Logistic Regression, K-Nearest Neighbors Classifier, Naive Gaussian Bayes, Bagging Algorithm, Gradient Boosting, Support Vector Linear Classification and (4) Evaluation of the model.

The work presented in [43] focused on the early prediction of type-2 diabetes with the use of effective techniques to reduce the mortality rate, using unsupervised deep neural networks. The authors used F1-score feature selection on the Pima Indian Dataset after the pre-processing phase to reduce noise and empty entries.

In the work of [33], a system has been proposed to explore predictive analytics for early detection of diabetes using several machine-learning algorithms, such as K-Nearest Neighbors, Logistic Regression, Support Vector Machine, Naive Bayesian, Random Forest, Decision Tree on Pima Indian Dataset which contains 768 patients from an Indian population.

In order to identify misdiagnosed type-1 diabetes patients from patients with a prior type-2 diabetes diagnosis, the XGBoost machine learning model has been used in [34]. The model utilized a total of 932 variables, which were initially broken down into various metrics for prediction. Following recursive feature elimination (RFE), the model achieved an optimal balance between precision and complexity by selecting the top 250 variables. These 250 variables were then used to generate the model's outputs.

In order to predict *Diabetes mellitus* in Nigeria at an early stage, a new approach has been proposed

in [35]. To accomplish their objective, authors collect data *via* oral interviews with patients and performed pre-processing and transformation using label encoder and standard scaler, respectively. After that, three supervised learning algorithms (K-Nearest Neighbors, Decision Trees and Artificial Neural Networks) were applied on nine attributes in order to find the best one.

The aim of the work presented in [36] is to employ several machine-learning techniques in order to explore age adaptation in predicting the risk of *Diabetes mellitus*. The used machine-learning algorithms are: Linear Regression, Logistic Regression, Polynomial Regression, Neural Network, Support Vector Machines, Random Forest and XGboost. To examine the impact of age, the authors divided the dataset into six age groups, evenly distributed, where individuals aged 81 were excluded from the analysis.

The authors in [37] derived several datasets from PIMA dataset by applying various pre-processing techniques in order to implement four machine-learning algorithms for diabetes prediction. The used algorithms are: K-Nearest Neighbors, Decision Tree, Random Forest and Support Vector Machines.

Given the advantages offered by ontologies, the authors of [39] proposed an approach based on ontologies and machine-learning algorithms. After pre-processing and cleaning the PIDD dataset, six machine-learning algorithms; namely, SVM, KNN, ANN, Naive Bayes, Logistic Regression and Decision Tree, were tested in order to choose the best one. In parallel, they extracted generated rules from the Decision Tree algorithm and imported them to Protegé using the SWRLTab plugin.

A new dataset was created in [40], where new diabetes risk factors were used. The authors followed five steps that are: Dataset Collection, Data Pre-processing, Clustering, Model Creation and Evaluation. In the Clustering step, the K-means algorithm was applied on the most correlated attributes (glucose and age) in order to classify each patient into diabetic or non-diabetic. For the model creation step, the following algorithms were tested: Support Vector Classifier, Random Forest Classifier, Decision Tree Classifier, Extra Tree Classifier, Ada Boost algorithm, Perceptron, Linear Discriminant Analysis algorithm, Logistic Regression, K-Nearest Neighbour, Gaussian Naïve Bayes, Bagging algorithm and Gradient Boost Classifier.

An ontology-based machine learning was proposed in [41] in order to predict diabetes. To achieve their goal, the authors followed three phases that are: Data pre-processing, Creation of the Decision Tree by using J4.8 classifier algorithm in Weka and finally the creation of the ontology. After creating the ontology and importing the dataset, the SWRL rules were extracted from the Decision Tree for reasoning. Then, Pellet reasoner was applied to infer whether the patient is diabetic or not.

In order to predict diabetes using machine-learning algorithms on Hadoop cluster, a new architecture is proposed in [42]. This architecture includes three modules that are: Diabetes Data Collection Module, Diabetes Prediction Module and Prediction Results Module. The first module is responsible for the collection of data from different hospitals. The second module is composed of different sub-modules that are: Features extraction, Features selection by using Information gain method and Hadoop Cluster that apply several machine-learning algorithms (Neural network, Support vector machine, Decision tree, Naive Bayes and Random forest).

In order to predict gestational *Diabetes mellitus* in Singapore, the authors of [43] combined coalitional game theory concepts with machine learning. Shapley values and the SHapley Additive exPlanations (SHAP) framework was combined with CatBoost tree ensembles for feature selection. Logistic Regression, Support Vector Machine, CatBoost Gradient Boosting and Multilayer Perceptron Artificial Neural Network were applied on the collected dataset.

In the same context, a machine-learning model was proposed in [44] in order to prevent the progression of gestational *Diabetes mellitus* to type-2 diabetes. For this purpose, Shapley values were combined with CatBoost tree ensembles to perform feature selection and Logistic Regression, Support Vector Machine, CatBoost Gradient Boosting and Artificial Neural Network were implemented and tested in order to choose the best algorithm. Table 1 presents a comparison among the presented works in terms of method and data.

Table 1. Comparison among the related works.

Work	Used dataset	Data size	Selected variables	Class	Limits
[27]	New	112	Data from CGM, meal, medication, insulin, physical activity, quality of sleep and glucose	Type-1 Diabete	Number of data
[28]	Ohio T1DM dataset	6	Glucose level, insulin, carbohydrate intake and mass index	/	The high rate of CGM missing values and the training-set length
[29]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[30]	New	158	Gender, BMI, Family history of diabetes, Blood pressure, duration of diabetes suffers, Blood-glucose level, patient complication diseases	Retinophaty, Neurophaty and Nephrophaty	Number of data
[40]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[43]	New	800	Age, genetic function, pregnancy number, glucose, blood pressure, skin thickness, BMI, insulin, type of labor and output	Diabetic and Non-diabetic	Number of data
[33]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[34]	IQVIA Ambulatory Electronic Medical Records	737,776	250	Type-1 and type 2 diabetes	The use of one algorithm
[35]	New	255	Age, sex, number of pregnancies, glucose level, blood pressure level, body mass index, height, weight and how regularly they exercise	Diabetic and Non-diabetic	Number of data
[36]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[37]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[39]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[40]	New	800	Number of Pregnancies, Glucose Level, Blood Pressure, Skin Thickness(mm), Insulin, BMI, Age, Job Type(Office-work/Fieldwork/Mac work) and Outcome	Diabetic and Non-diabetic	Number of data

[41]	PIDD	768	Pregnancies, Age, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Class	Diabetic and Non-diabetic	The performance is adversely affected by missing or unknown values in the datasets
[42]	National Institute of Diabetes	75,664	13 variables	Diabetic and Non-diabetic	The number of node hadoop cluster
[43]	New	909	Prepregnancy obesity, family history of diabetes, previous history of GDM, previous delivery of a macrosomic baby and Indian ethnicity	Diabetic and Non-diabetic	Number of data
[44]	New	561	Demographics, medical or obstetric history, physical measures, lifestyle information and GDM diagnosis	Diabetic and Non-diabetic	Number of data

3.1 Our Contribution

The objective of this work is to propose an automatic diabetes-prediction system, in order to raise patient awareness and avoid serious complications that can occur in the long term. Our system makes it possible to identify people at high risk of developing diabetes in order to put in place early preventive measures and slow down the progression of the disease, which results in the improvement of the quality of life. We can summarize our contribution in the following points:

- The construction of our own dataset from scratch by collecting Algerian data. The goal behind this initiative is to collect the data that influences the diabetes disease in Algeria and to focus on the risk factors of this last. Creating our own dataset was a crucial step, because the publically available datasets don't represent all the important risk factors of diabetes; so, they do not reflect the reality of the disease. For this, we have collected all the features that concern the patient; namely, his/her demographic, anthropometric, genetic, lifestyle, medical and laboratory information.
- The application of pre-processing techniques to improve the quality of the collected medical data,
- Exploratory data analysis with the intention of identifying the most significant features and the correlation between them. To do that, we applied several feature-extraction techniques, such us: ANOVA, RFE and the features proposed by the PIDD dataset.
- Application of several machine-learning algorithms, to compare the obtained results and choose the best among them,
- Drawing possible predictive conclusions in terms of the obtained results and deciding whether the patient is diabetic or non-diabetic.

4. PROPOSED APPROACH

Diabetes mellitus is a major and prevalent pathology that affects many people. Several factors can increase the risk of developing *Diabetes mellitus*, such as age, obesity, physical inactivity, family history, lifestyle, unbalanced diet, high blood pressure, …etc. Figure 1 presents the architecture of our diabetes prediction system.

In order to predict the diabetes disease, we will follow the following steps:

4.1 Data Collection

Data collection is a crucial step in any scientific research study. In our case, we collected real data in Algeria, in the city of Oran, from diabetic and non-diabetic patients, using a questionnaire administered at the Oran University hospital. The purpose was to collect the largest amount of data on these patients.

The questionnaire was validated by experts in the field in order to guarantee its quality and relevance. The questions asked were adapted to the studied population and allowed an in-depth analysis of the various risk factors associated with diabetes. The answers obtained were carefully processed and

used to feed our diabetes-prediction system.

Our internship took place in two departments; namely, the Internal Medicine department reserved for diabetic patients and the Endocrinology department to collect data from non-diabetic patients affected by other conditions that may cause diabetes. On the other hand, we created and shared an online questionnaire *via* social-media networks (Facebook, Twitter, Linkedin, …etc.), to gather more data and enrich our dataset. We have implemented rigorous protocols to guarantee the quality of the collected data, in particular by ensuring the confidentiality and anonymity of patients. Table 2 shows the collected data, with an indication of the start and end period of the collection and the total number of patients taken.

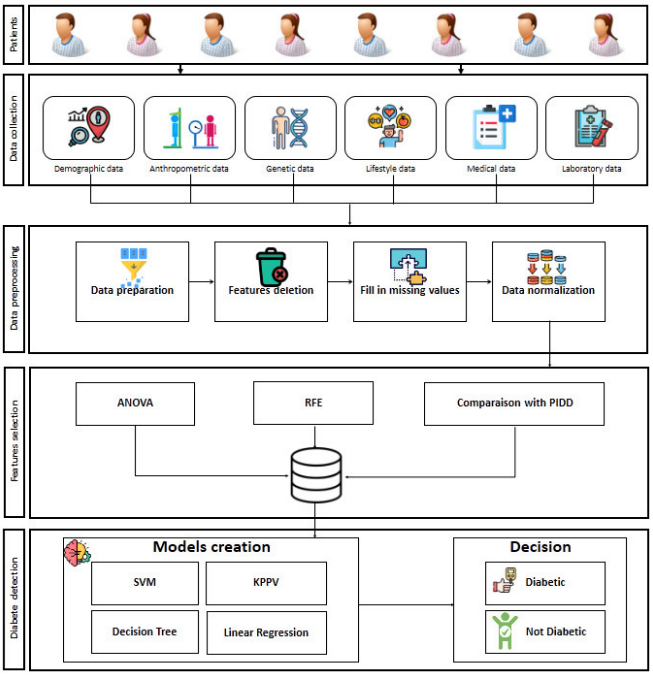


Figure 1. Proposed architecture.

Table 2. Our dataset.

Start date of data collection	End date of data collection	Dataset size	Number of diabetic patients	Number of non-diabetic patients
09 February 2023	17 May 2023	823	418	405

The general population statistics are presented in Tables 3, 4 and 5.

Table 3. Population description.

Features	**Min.**	**Max.**	**Average**	**Mode**	**Median**
Age	16	82	42	21	33
Height	120	190	164	160	165
Weight	44	160	71	65	71

Table 4. Distribution of the population according to sex.

Women	65%
Men	35%

Table 5. Number of individuals with specific diseases.

Disease	**Non-diabetic**	**Diabetic**	**Total**
Hypertension	188	126	314
Thyroid	52	10	62
Heart disorder	74	50	124
Osteoarthritis	34	17	51

Allergies	102	0	102
Bone problem	15	0	15
Cholesterol	25	18	43
Sight problem	56	23	79
Asthma	29	0	29
Anemia	57	36	93
Depression	34	51	85
Kidney problem	27	28	55
Headaches	42	0	42
Weakness	0	36	36

We divided the collected features into 6 categories, that are:

1) **Demographic data:** The patient demographic data includes: city, age, sex, family situation, profession, state of pregnancy, number of pregnancies and number of children,
2) **Anthropometric data:** The patient anthropometric data includes 4 characteristics, that are: size, weight, BMI and obesity.
3) **Genetic data:** The genetic data includes: familial genetic load of diabetes and diabetes family tree function.
4) **Lifestyle data:** The lifestyle data comprises: sport, anxious nature, type of consumed sugar, balanced diet, consumption of fruits and vegetables, consumption of fast food, fasting rhythm, smoking and alcohol.
5) **Medical data:** The medical data encompasses: diagnosis of diabetes during pregnancy, history of health problems, the type of disease, antihypertensive medication, hypoglycemia diagnosis, diabetes, duration, type of diabetes, treatment, health problems and complications after diabetes and the type of complication disease.
6) **Laboratory data:** The data related to patients' blood tests includes 12 features, which are: Blood sugar (in g/l), Total cholesterol (in g/l), Urea (in g/l), HbA1c (in percentage (%)), Triglycerides (in g/l), Creatinine (in mg/l), TGO/ASAT (in U/l), TGP/ALAT (in U/l), TSH/TSHus (in U/l), T4/FT4 (in pg/ml), T3/FT3 (in pg/ml) and Uric Acid (in mg/l).

4.2 Data Pre-processing

After describing all the data that we have collected, we will proceed to data visualization and analyze the different characteristics, then we move on to cleaning and correcting the records to obtain reliable data, ready to be used for diabetes prediction.

The goal of this phase is to improve the quality of the collected data, eliminate errors and deal with inconsistencies and missing data, to make more informed and accurate decisions. In order to prepare our data for training predictive models, we will follow the following steps:

4.2.1 Data Preparation

The first step is the detection and correction of the typing errors, as they are common in medical data. Next, we transformed all the categorical data into numerical data, such as Pregnant = 1 and Not-pregnant = 0 or Single = 0, Married = 1, Divorced = 2 and Widowed = 3. We also removed records that have empty values in most columns.

4.2.2 Feature Deletion

In this step, we decided to delete some column in order to take only the relevant features. For this, we visualized the missing values in our dataset represented by the white color in Figure 2. We notice that the data of the blood tests has a very high rate of missing values, such as FT3, FT4, THus, …etc.

After missing-data visualization, we made a percentage of the missing values for each attribute in order to categorize them into 3 groups, as follows:

- Group 1: Features that have between 0% and 40% missing values,
- Group 2: Features that have between 40% and 80% missing values,
- Group 3: Features that have between 80% and 100% missing values.

After that, we removed group-3 attributes, as they have a large percentage of missing values. We deleted also the irrelevant features, like the city attribute (it is specific to the geographical location and may not provide a direct correlation with the diabetes-prediction study). We removed also height and weight (they have a correlated relationship with the BMI column).

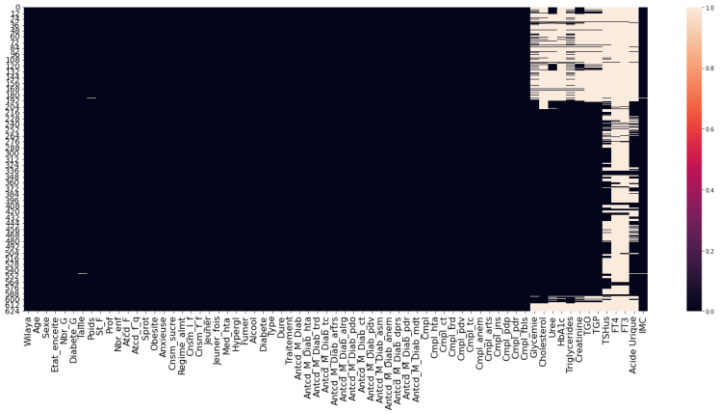


Figure 2. Visualization of missing data.

4.2.3 Fill in Missing Values

To fill in the missing values, we replaced the missing data with the value 0 or the median value according to the attribue. The median consists of replacing missing or outlying values in the data with the median of the entire dataset. It is a robust measure of the central tendency of the data that is not influenced by extreme values or outliers.

4.2.4 Data Normalization

We carried out data normalization to put all the variables on the same scale. This ensures that certain variables do not dominate others when building the model.

4.3 Feature Selection

Feature selection is the process of selecting a sub-set of the most relevant features in the dataset to describe the target variable. It improves computation time, generalization performance and interpretation problems. There are several types of feature-selection techniques, in our case, we used:

4.3.1 Uni-varied Selection

This technique consists in evaluating each characteristic independently of the other characteristics by estimating the correlation or the dependence between its characteristics and the target variable, in order to choose the variables that have the strongest correlation or target dependence. Among these univariate selection techniques, we have the ANOVA method.

ANOVA (ANalysis of Variance) is a well-known statistical method to determine whether there is a difference in means between two groups. In our study, the ANOVA test was used to select the significant numerical characteristics in the prediction of diabetes. The F-statistic is used for feature ranking. The larger the value of the F-statistic, the better the discriminative ability of the feature [38]. The F-value is calculated as follows (see formula 1):

$$F = \frac{SSB}{df_b} \div \frac{SSW}{df_w} \tag{1}$$

where SSB (sum of squares between groups) is the variation of the means of the groups from the total grand mean and SSW (sum of squares within the groups) is the sum of the squared deviations from the means of the groups and at each sighting. The degrees of freedom for the mean square between and within groups are defined by df_b and df_w, respectively. For all numerical features in the dataset, the F-value was calculated using the previous equation 1 and the features with the highest values were selected [38].

Our goal was to assess the importance of each feature in our database based on its contribution to the variance in the data, so that we could use this information to select the most important features. The

best ANOVA selection is 32 out of 68 features (see Figure 3).

The visualization of the selected features and the F-score provide a visual overview of the most important variables for prediction, thereby helping better understand the influence of each feature on the final outcome (see Figure 4).

- Age
- Nbr_G
- Diabete_G
- St_F
- Prof
- Nbr_enf
- Atcd_f_q
- Sprot
- Obesite
- Cnsm_l_f
- Jeuner_fois
- Med_hta
- Hypergl
- Fumer
- Antcd_M_Diab
- Antcd_M_Diab_hta
- Antcd_M_Diab_tc
- Antcd_M_Diab_ct
- Cmpl
- Cmpl_hta
- Cmpl_ct
- Cmpl_tc
- Cmpl_anem
- Cmpl_pdp
- Cmpl_pdr
- Cmpl_fbls
- Glycemie
- HbA1c
- Triglycerides
- Creatinine
- TGP
- Acide Urique

Figure 3. The features selected by ANOVA.

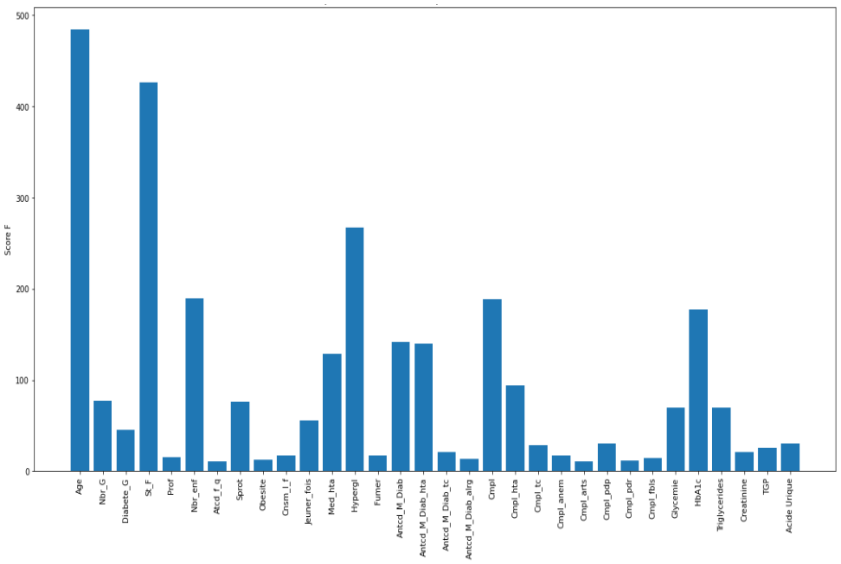


Figure 4. The importance of the features selected by ANOVA and the F-score.

4.3.2 Recursive Feature Elimination (RFE)

Recursive feature elimination is a recursive procedure to select features based on model accuracy. The metric determines the discriminative ability of the features. At each iteration, the rank score metric is calculated and lower-ranked features are eliminated. The recursive procedure is repeated until the desired number of features is reached. In this study, RFE was used as the final step in the feature- selection procedure [38].

We could use recursive feature elimination (RFE) to identify the most important features one by one. We used this technique with the logistic regression algorithm to evaluate their performance on the selected characteristics. According to this method, the optimal number of data selections is 20 among 68 features (see Figure 5).

The visualization of features selected by RFE shows the positive and negative relationships depending on the effect of each variable on the target variable (see Figure 6).

- Etat_enceite
- Diabete_G
- St_F
- Prof
- Atcd_f_q
- Cnsm_sucre
- Cnsm_l_f
- Jeuner_fois
- Hypergl
- Fumer
- Antcd_M_Diab
- Antcd_M_Diab_hta
- Antcd_M_Diab_alrg
- Antcd_M_Diab_asm
- Antcd_M_Diab_pdr
- Antcd_M_Diab_mdt
- Cmpl
- Cmpl_pdp
- Cholesterol
- HbA1c

Figure 5. The features selected by recursive feature elimination (RFE).

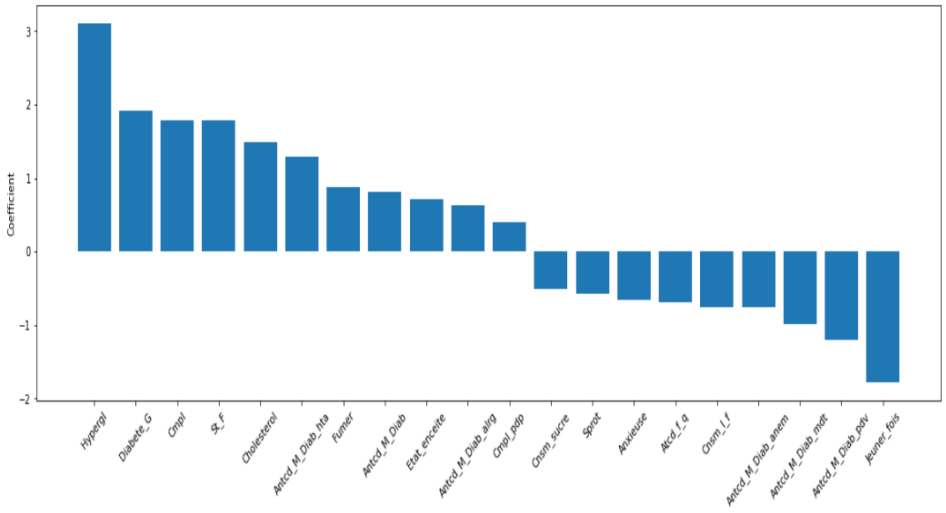


Figure 6. The importance of the features selected by recursive feature elimination.

4.3.3 Comparison with the Pima Indian Diabetes Dataset

The Pima Indian Diabetes Dataset (PIDD) is a well-known dataset used for diabetes prediction. It contains medical information from 768 female Pima Indians, including 9 features, such as glucose level, blood pressure, BMI and age. The target variable indicates whether the individual has diabetes (1) or not (0). This dataset is widely used in machine learning and data-mining research for predicting the risk of diabetes based on the given features.

It is important to note that this dataset is imbalanced, as there are significantly more non-diabetic samples compared to diabetic samples (500 samples belonging to non-diabetic persons and 268 samples belonging to diabetic persons).

In the dataset, six attributes contain zero values for some samples. The occurrences of zero values in these attributes are as follows: 111 for pregnancy, 5 for glucose, 35 for blood pressure, 227 for skin thickness, 374 for insulin and 11 for BMI. The presence of these zero values might be due to errors during data collection or missing data.

These zero values can potentially impact the performance of the classifier negatively. To create a reliable prediction model, it is essential to handle these missing values appropriately and fill them with suitable values. This process ensures that the model can make accurate predictions and avoid any potential biases caused by incomplete or incorrect data.

For a comparative analysis, we compared our database with the famous Pima Indians Diabetes Dataset, which is widely used in the scientific literature for the study of diabetes. This approach will allow us to evaluate and improve the study to fully understand the differences and similarities between the data collected in Algeria and the reference data in the scientific literature. So, we selected the same available features in our dataset according to PIDD. Table 6 shows a summary of the selected features according to the PIMA dataset.

Table 6. Summary of the selected futures according to the PIMA dataset.

Pima Indian Diabetes Dataset	**Our Dataset**
Glucose	Blood sugar
Pregnancies	Number of pregnancies
Blood pressure	Antihypertensive medication
Skin thickness	/
Insulin	/
BMI	BMI
Diabetes pedigree Function	Diabetes family tree function
Âge	Âge
Outcome	Diabete

Figure 7 presents the correlation matrix of the selected features.

Figure 7. Ccorrelation matrix of the selected features.

4.4 Diabetes Prediction

We split the dataset into 70% for the training set and 30% for the test set. The training phase is a crucial part in the creation of any machine-learning and deep-learning model, including for diabetes prediction, in order to optimize the parameters so that the model is capable of improving performance. For the creation of the model, we used several machine-learning algorithms to evaluate their performance and choose the most robust one. This step also includes finding the best hyper-parameters for each algorithm and training the model on the training data. Once the model is trained, it can be used to predict output values for new data.

The algorithms used in this step are: SVM, KNN, Decision Tree and Logistic Regression. Once the models are trained, they are saved to use them in the test phase or to deploy them to predict new cases of patients.

The next phase is testing and evaluation, which is an important step in the prediction process, because it allows to evaluate the performance of the model on the test data, it includes several steps, which are:

1) Retrieving test data,
2) Deploying the machine-learning model from the database,
3) Evaluating performance by estimating performance metrics, such as accuracy, precision, …etc.
4) Analyzing the results to understand the strengths and weaknesses of the created model. This may include prediction visualizations,
5) Improving the model created from the previous analysis in order to adjust the hyperparameters again or make modifications in the previous steps. This phase can be repeated until the model reaches high performance.

5. EXPERIMENTATIONS

To carry out our experiments and train our prediction models, we used a single personal computer with the following characteristics (see Table 7):

Table 7. The specifications of the used machine.

PC	CPU	RAM	GPU	Storage space
DELL	Intel® Core™ i7-8550U CPU	8.00 Go	Mesa Intel® UHD Graphics 620 (KBL GT2)	1 TB

We have implemented several machine-learning algorithms, such as SVM, KNN, Decision Tree and Logistic Regression. Our objective was to compare these algorithms with each other and select the one that offers the best performance in terms of recall, precision, accuracy, F1-measurement, RMSE, MSE and MAE. In addition, we will examine the results obtained by each algorithm on a dataset, using 3 feature-extraction techniques; namely, Anova, RFE and comparison with the PIMA dataset.

5.1 SVM

The results obtained with the SVM algorithm are presented in Table 8, highlighting the performance of the model.

Table 8. Diabetes-prediction results with the SVM algorithm.

	Accuracy	Recall	Precision	F1- measure	RMSE	MSE	MAE
ANOVA	96%	96.97%	95.52%	96.24%	0.2	0.04	0.024
RFE	97.60%	100%	95.65%	97.78%	0.16	0.024	0.024
Comparison with PIMA	89.6%	84.85%	94.92%	89.6%	0.32	0.10	0.09

After analyzing the results, we found that the use of the SVM algorithm for the prediction of diabetes does not show a large significant difference in accuracy between the use of RFE and ANOVA. The difference observed is minimal, with only 1% variation between the two techniques.

5.2 KNN

The results obtained with the KNN algorithm are presented in Table 9.

Table 9. Diabetes-prediction results with the KNN algorithm.

	Accuracy	Recall	Precision	F1- measure	RMSE	MSE	MAE
ANOVA	97.60%	96.97%	98.49%	97.71%	0.16	0.02	0.02
RFE	95.20%	92.42%	98.39%	95.31%	0.22	0.05	0.05
Comparaison with PIMA	92.80%	90.91%	95.24%	93.02%	0.27	0.07	0.07

From the results presented in Table 9, it is clear that using the KNN algorithm in combination with the Anova method for feature selection gives superior performance compared to other feature selection techniques. Indeed, the model achieves an accuracy of 97.60%, indicating its ability to accurately predict diabetes.

5.3 Decision Tree

The results obtained with the decision-tree algorithm are presented in Table 10.

Table 10. Diabetes-prediction results with the decision-tree algorithm.

	Accuracy	Recall	Precision	F1- measure	RMSE	MSE	MAE
Anova	95.20%	93.94%	96.88%	95.38%	0.22	0.05	0.05
RFE	97.60%	98.48%	97.01%	97.74%	0.16	0.02	0.02
Comparaison with PIMA	93.60%	90.91%	96.77%	93.75%	0.25	0.06	0.06

The decision-tree algorithm produced promising results in our study. When we used the features extracted by RFE, we obtained an accuracy of 97.60%. However, using features selected by ANOVA, the accuracy decreased, reaching 95.20%.

5.4 Logistic Regression

The results obtained with the logistic-regression algorithm are shown in Table 11.

Table 11. Diabetes-prediction results with the logistic-regression algorithm.

	Accuracy	Recall	Precision	F1- measure	RMSE	MSE	MAE
Anova	96%	93.94%	98.41%	96.12%	0.20	0.04	0.04
RFE	97.60%	100%	95.65%	97.78%	0.15	0.02	0.02
Comparaison with PIMA	90.40%	83.33%	98.21%	90.16%	0.31	0.09	0.09

The results obtained demonstrate that there are no significant differences between the used feature-selection methods. However, it is important to note that the RFE algorithm achieved an accuracy of 97.60% and this is the best-performing method.

5.5 Discussion

To facilitate the understanding and analysis of the results, we have divided the results into three separate tables, corresponding to each feature-selection method used in our study. This allows us to compare the performances of different methods and identify the best combination of algorithm and future-selection method for diabetes prediction.

Each table presents the key performance measures, such as precision, recall and F-measure, obtained for each combination of algorithm and feature-selection method. By examining these tables, we will be able to evaluate the performance of each selection method and determine which one offers the best results in terms of predicting diabetes.

Table 12. Results of the ANOVA selection method for the prediction of diabetes.

	SVM	KNN	Decision Tree	Logistic Regression
Accuracy	96%	97.60%	95.20%	96%
Recall	96.97%	96.97%	93.94%	93.94%
Precision	95.52%	98.49%	96.88%	98.41%
F1-measure	96.24%	97.71%	95.38%	96.12%
RMSE	0.2	0.16	0.22	0.20
MSE	0.04	0.02	0.05	0.04
MAE	0.024	0.02	0.05	0.04

Table 13. Results of the RFE selection method for the prediction of diabetes.

	SVM	KNN	Decision Tree	Logistic Regression
Accuracy	97.60%	95.20%	97.60%	97.60%
Recall	100%	92.42%	98.48%	100%
Precision	95.65%	98.39%	97.01%	95.65%
F1-measure	97.01%	95.31%	97.74%	97.78%
RMSE	0.16	0.22	0.16	0.15
MSE	0.024	0.05	0.02	0.02
MAE	0.024	0.05	0.02	0.02

Table 14. Results of the comparative method with PIMA for the prediction of diabetes.

	SVM	KNN	Decision Tree	Logistic Regression
Accuracy	89.6%	92.80%	93.60%	90.40%
Recall	84.85%	90.91%	90.91%	83.33%
Precision	94.92%	95.24%	96.77%	98.21%
F1-measure	89.6%	93.02%	93.75%	90.16%
RMSE	0.32	0.27	0.25	0.31
MSE	0.10	0.07	0.06	0.09
MAE	0.09	0.07	0.06	0.09

From the results of the previous tables, we can identify the best machine-learning algorithm for diabetes prediction. The most efficient algorithms are:

- The KNN algorithm with the ANOVA feature-extraction method. When we use the ANOVA method, we obtain an accuracy rate of 97.60%.
- The Logistic Regression algorithm performs well, using the RFE feature-extraction method. When we apply the RFE method, we obtain an accuracy rate of 97.60%.
- By using the SVM algorithm with the RFE method, we manage to obtain a remarkable accuracy rate of 97.60%,
- The Decision Tree algorithm, combined with the RFE method for feature selection, provides an accuracy rate of 93.60%.

We note that during the experiments, we sought to optimize the performance of our algorithms by carrying out exhaustive tests with different parameter configurations. The objective was to find the parameters that would allow us to obtain the most efficient results. The results of each parameter configuration were carefully noted and compared. As an example, we set the number of clusters in the KNN algorithm to 7 and used Euclidean distance as a similarity measure. In the SVM algorithm, we chose kernel = 'linear'. For the decision tree, we took entropy as a division criterion.

In order to evaluate the performance of our classification models, in particular with regard to their abilities to distinguish between positive and negative classes, we plotted the ROC (Receiver Operating Characteristic) curve and the area under the curve (AUC) which are presented in Figure 8. These metrics are commonly used in the evaluation of classification models, particularly in the field of diabetes prediction, where the distinction between true positives and false positives is of critical importance. From the presented results, the AUC of our models is closer to 1.0 which is considered efficient.

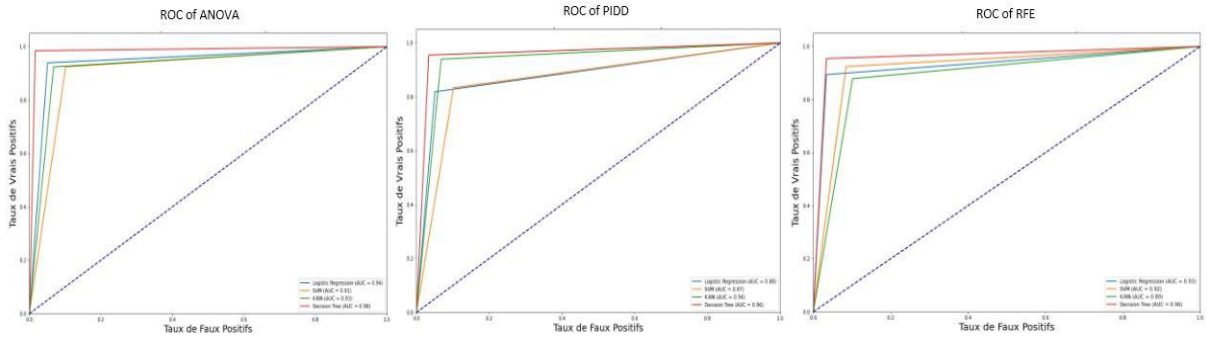


Figure 8. The ROC and AUC results.

The limitation of this work lies in the number of collected data, since the volume of data is a critical factor in data analysis and modeling projects, especially in the medical field. Due to time, resource and data-access constraints, we had to work with a limited sample of data. This may lead to potential biases in our results, as a larger sample could have provided a more complete and representative view of the problem.

6. CONCLUSION

Diabetes prediction has become a growing area of interest due to the increasing prevalence of diabetes, advances in technology and the growing importance of health data. Research efforts in this area aim to develop accurate predictive tools that can aid in the prevention, early diagnosis and effective management of diabetes, thereby improving the health and well-being of individuals affected by this disease.

In this work, we implemented and evaluated several algorithms, such as SVM, KNN, Logistic Regression and Decision Tree, by comparing their performances and analyzing the obtained results. Our results demonstrated that the KNN algorithms proved to be particularly effective in the prediction of diabetes, because it took advantage of its nearest neighbor-based approach to achieve good results. It is also important to emphasize the importance of feature selection in the prediction of diabetes. We found that different selection methods influenced the performance of the algorithms,

highlighting the importance of a well-considered approach, such as RFE or ANOVA, to choose the most relevant features.

In terms of future perspectives, there are several aspects to consider to improve and further develop the diabetes-prediction system:

- In terms of the dataset, it would be interesting to extend our study to large data on other wilayas to reach the entire Algerian population,
- Collecting other risk factors for diabetes,
- Developing a useful online website or mobile application and deploying it on Play store.

REFERENCES

[1] S. M. H. Mahmud et al., "Machine Learning Based Unified Framework for Diabetes Prediction," Proc. of the 2018 Int. Conf. on Big Data Engineering and Technology (BDET 2018), pp. 46–50, 2018.

[2] A. Singh, A. Dhillon, N. Kumar, M. S. Hossain, G. Muhammad and M. Kumar, "eDiaPredict: An Ensemble-based Framework for Diabetes Prediction," ACM Transactions on Multimedia Computing, Communications and Applications, vol. 17, no. 2, pp 1–26, 2021.

[3] L. Xu, J. He and Y. Hu, "Early Diabetes Risk Prediction Based on Deep Learning Methods," Proc. of the 4th Int. Conf. on Pattern Recognition and Artificial Intelligence, pp. 282-286, Yibin, China, 2021.

[4] M. Belhadj et al., "BAROMÈTRE Algérie: Enquête Nationale sur la prise en Charge des Personnes Diabétiques," Médecine des Maladies Métaboliques, vol. 13, no. 2, pp. 188-194, 2019.

[5] D. S. Sisodia and R. Agrawal, "Data Imputation-based Learning Models for Prediction of Diabetes," Proc. of the 2020 Int. Conf. on Decision Aid Sciences and Application (DASA), pp. 966-970, 2020.

[6] H. Song and S. Lee, "Implementation of Diabetes Incidence Prediction Using a Multilayer Perceptron Neural Network," Proc. of the IEEE Int. Conf. on Bioinformatics and Biomedicine, pp. 3089-3091, Houston, USA, 2021.

[7] Z. Punthakee, R. Goldenberg and P. Katz, "Definition, Classification and Diagnosis of Diabetes, Prediabetes and Metabolic Syndrome," Canadian Journal of Diabetes, vol. 42, no. 1, pp. 10-15, 2018.

[8] Y. Wei, W. Guo, B.W. Ling, Y. Dai and Q. Liu, "Both Forward Approach and Backward Approach for Performing Both Regressions and Classifications Using the Histogram Information for Predicting the Baseline Screening Scores for Performing the Prognostic of the Diabetes," Signal, Image and Video Processing, vol. 17, pp. 3803–3809, 2023.

[9] A. Al-Sideiri, Z. B. C. Cob and S. B. M. Drus, "Machine Learning Algorithms for Diabetes Prediction: A Review Paper," Proc. of the 2019 Int. Conf. on Artificial Intelligence, Robotics and Control (AIRC '19), pp. 27–32, 2019.

[10] M. Komi, J. Li, Y. Zhai and X. Zhang, "Application of Data Mining Methods in Diabetes Prediction," Proc. of the 2017 2nd IEEE Int. Conf. on Image, Vision and Computing (ICIVC), pp. 1006-1010, Chengdu, 2017.

[11] A. H. Khan and J. E. Pessin, "Insulin Regulation of Glucose Uptake: A Complex Interplay of Intracellular Signalling Pathways," Diabetologia, vol. 45, pp. 1475–1483, 2002.

[12] S. V. Hemanth, S. Alagarsamy and T. D. Rajkumar, "Convolutional Neural Network-based Sea Lion Optimization Algorithm for the Detection and Classification of Diabetic Retinopathy," Acta Diabetologica, vol. 60, pp. 1377–1389, 2023.

[13] X. Li, M. Curiger, R. Dornberger and T. Hanne, "Optimized Computational Diabetes Prediction with Feature Selection Algorithms," Proc. of the 2023 7th Int. Conf. on Intelligent Systems, Metaheuristics and Swarm Intelligence (ISMSI '23), pp. 36–43, DOI: 10.1145/3596947.3596948, 2023.

[14] M. M. Hassan, Z. J. Peya, S. Mollick, M. A. Billah, M. M. Hasan Shakil and A. U. Dulla, "Diabetes Prediction in Healthcare at Early Stage Using Machine Learning Approach," Proc. of the 12th Int. Conf. on Computing Communication and Networking Technologies, pp. 01- 05, Kharagpur, India, 2021.

[15] J. M. Ekoé, Z. Punthakee, T. Ransom, A. P. H. Prebtani and R. Goldenberg, "Dépistage du Diabète de Type 1 et de Type 2," Canadian Journal of Diabetes, vol. 37, pp. S373-S376, 2013.

[16] P. J. Shermila, A. Ahilan, M. Shunmugathammal and J. Marimuthu, "DEEPFIC: Food Item Classification with Calorie Calculation Using Dragonfly Deep Learning Network," Signal, Image and Video Processing, vol. 17, pp. 3731–3739, 2023.

[17] S. Brahimi and Y. F. Drioua, Etude Rétrospective des Facteurs de Risques du Diabète au Niveau de l'EPSP EsSenia, M.Sc Thesis, University of Science and Technology of Oran Mohamed Boudiaf, USTO-MB oran, Algeria, 2021.

[18] F. Zafar et al., "Predictive Analytics in Healthcare for Diabetes Prediction," Proc. of the 2019 9th Int. Conf. on Biomedical Engineering and Technology (ICBET' 19), pp. 253–259, 2019.

[19] S. Mahajan, P. K. Sarangi, A. K. Sahoo and M. Rohra, "Diabetes Mellitus Prediction Using Supervised Machine Learning Techniques," Proc. of the 2023 Int. Conf. on Advancement in Computation and Computer Technologies, pp. 587-592, Gharuan, India, 2023.

[20] M. van der Schaar et al., "How Artificial Intelligence and Machine Learning Can Help Healthcare Systems Respond to COVID-19," Machine Learning, vol. 110, no. 1, pp. 1–14, 2021.
[21] M. H. Arnold, "Teasing out Artificial Intelligence in Medicine: An Ethical Critique of Artificial Intelligence and Machine Learning in Medicine," Journal of Bioethical Inquiry, vol. 18, no. 1, pp. 121–139, 2021.
[22] F. Ali et al., "An Intelligent Healthcare Monitoring Framework Using Wearable Sensors and Social Networking Data," Future Generation Computer Systems, vol. 114, pp. 23–43, 2021.
[23] P. Whig, K. Gupta, N. Jiwani, H. Jupalle, S. Kouser and N. Alam, "A Novel Method for Diabetes Classification and Prediction with Pycaret," Microsystem Technologies, vol. 29, pp. 1479-1487, 2023.
[24] B. Karaagac, K. M. Owolabi and E. Pindza, "A Computational Technique for the Caputo Fractal-fractional Diabetes Mellitus Model without Genetic Factors," Int. Journal of Dynamics and Control, vol. 11, pp. 2161–2178, 2023.
[25] E. Daniel, J. Johnson, U. A. Victor, G. V. Aditya and S. A. Sibby, "An Efficient Diabetes Prediction Model Using Machine Learning," Proc. of the 4th Int. Conf. on Electronics and Sustainable Communication Systems, pp. 1202-1208, Coimbatore, India, 2023.
[26] D. N. Katsarou et al., "Short Term Glucose Prediction in Patients with Type 1 Diabetes Mellitus," Proc. of the 44th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society, pp. 329-332, Glasgow, Scotland, United Kingdom, 2022.
[27] W. Gu, Z. Zhou, Y. Zhou, M. He, H. Zou and L. Zhang, "Predicting Blood Glucose Dynamics with Multi-time-series Deep Learning," Proc. the 15th ACM Conf. on Embedded Network Sensor Systems, pp. 1-2, DOI: 10.1145/3131672.3136965, 2017.
[28] T. Zhu, K. Li, P. Herrero, J. Chen and P. Georgiou, "A Deep Learning Algorithm For Personalized Blood Glucose Prediction," Proc. of the 27th Int. Joint Conf. on Artificial Intelligence and the 23rd European Conf. on Artificial Intelligence, pp. 1-5, 2018.
[29] A. Yahyaoui, A. Jamil, J. Rasheed and M. Yesiltepe, "A Decision Support System for Diabetes Prediction Using Machine Learning and Deep Learning Techniques," Proc. of the 1st Inte. Informatics and Software Engineering Conf. (UBMYK), pp. 1-4, Ankara, Turkey, 2019.
[30] C. Fiarni, E. M. Sipayung and S. Maemunah, "Analysis and Prediction of Diabetes Complication Disease Using Data Mining Algorithm," Procedia Computer Science, vol. 161, pp. 449–457, 2019.
[31] A. Mujumdar and V. Vaidehi, "Diabetes Prediction Using Machine Learning Algorithms," Procedia Computer Science, vol. 165, pp. 292-299, 2019.
[32] P. B. M. Kumar et al., "Type 2: Diabetes Mellitus Prediction Using Deep Neural Networks classifier," Int. Journal of Cognitive Computing in Engineering, vol. 1, pp. 55-61, 2020.
[33] Himanshi, A. Agarwal, Sidharth and K. Middha, "Prediction of Diabetes Using Machine Learning Algorithms," Int. Research Journal of Modernization in Engineering Technology and Science, vol. 3, no. 12, pp. 1778-1782, 2021.
[34] R. Cheheltani et al., "Predicting Misdiagnosed Adult-onset Type 1 Diabetes Using Machine Learning," Diabetes Research and Clinical Practice, vol. 191, p. 110029, 2022.
[35] A. E. Evwiekpaefe and N. Abdulkadir, "A Predictive Model for Diabetes Mellitus Using Machine Learning Techniques (A Study in Nigeria)," The African Journal of Information Systems, vol. 15, no. 1, pp. 1-21, 2023.
[36] Y. Su, C. Huang, W. Yin, X. Lyu, L. Ma and Z. Tao, "Diabetes Mellitus Risk Prediction Using Age Adaptation Models," Biomedical Signal Processing and Control, vol. 80, p. 104381, 2023.
[37] S. C. Gupta and N. Goel, "Predictive Modeling and Analytics for Diabetes Using Hyper-parameter Tuned Machine Learning Techniques," Procedia Computer Science, vol. 218, pp. 1257–1269, 2023.
[38] H. M. Deberneh and I. Kim, "Prediction of Type 2 Diabetes Based on Machine Learning Algorithm," Int. Journal of Environmental Research and Public Health, vol. 18, no. 6, 2021.
[39] H. El Massari, Z. Sabouri, S. Mhammedi and N. Gherabi, "Diabetes Prediction Using Machine Learning Algorithms and Ontology," J. of ICT Standardization, vol. 10, no. 02, pp. 319–338, 2022.
[40] A. Mujumdar and V. Vaidehi, "Diabetes Prediction Using Machine Learning Algorithms," Procedia Computer Science, vol. 165, pp. 292–299, 2019.
[41] H. EL Massari, S. Mhammedi, Z. Sabouri and N. Gherabi, "Ontology-based Machine Learning to Predict Diabetes Patients," Proc. of the Int. Conf. on Information, Communication and Cybersecurity (ICI2C 2021), pp. 437–445, DOI: 10.1007/978-3-030-91738-8_40, 2022.
[42] N. Yuvaraj and K. R. SriPreethaa, "Diabetes Prediction in Healthcare Systems Using Machine Learning Algorithms on Hadoop Cluster," Cluster Computing, vol. 22, no. 1, pp. 1–9, 2019.
[43] M. Kumar et al., "Population-centric Risk Prediction Modeling for Gestational Diabetes Mellitus: A Machine Learning Approach," Diabetes Research and Clinical Practice, vol. 185, no. 01, pp. 01-11, 2022.
[44] M. Kumar et al., "Machine Learning-derived Prenatal Predictive Risk Model to Guide Intervention and Prevent the Progression of Gestational Diabetes Mellitus to Type 2 Diabetes: Prediction Model Development Study," JMIR Diabetes, vol. 07, no. 03, pp. 01-12, 2022.

ملخص البحث:

مرض السُّكَّر واحدٌ من أكثرِ الأمراضِ انتشاراً على مستوى العالم، كما أنّ معدل انتشاره آخذٌ في الارتفاع. والعواملُ التي تتسبّب في ذلك الانتشار المتزايد للمرض إنّما تتعلّقُ بالتّغذية ونمط الحياة من جانبٍ وبعواملٍ وراثيةٍ من جانبٍ آخر، الأمر الذي يشكل مُعضلة صحية عامة. لذا فإنّ الكشْف المبكّر عن المرض أمرٌ حاسِمٌ حتى يتمّ التّدخُّل لعلاجه بسرعةٍ وبالتّالي الحدُّ من تفاقمه وانتشاره.

يهدف هذا البحث الى اقتراح نظامٍ أوتوماتيكي لتوقُّع الإصابة بمرض السُّكَّر، باستخدام عددٍ من تقنيات تعلُّم الآلة. وعن طريق جمْع بياناتٍ عن المرض في البيئة الجزائرية، تمّ استخدام عددٍ من طُرق انتقاء السِّمات ومقارنة أداء كلٍّ منها بالسِّمات المقترحة من قبل مجموعة البيانات الهندية الخاصّة بمرض السُّكَّر، المعروفة باسم PIMA.

وقد أسفرت نتائج البحث عن معلوماتٍ قيّمةٍ حولَ المقارنة بين تقنيات تعلُّم الآلة المختلفة المستخدمة من حيث الأداء المرتبط بتوقُّع المرض وكذلك حول أهمّية السِّمات المختارة التي تمّت الاستفادة منها في النّموذج المقترح في هذه الدراسة.

JJCIT Annual List of Reviewers (2023)

Name, *Affiliation*, Country

ABD Amir Hamzah, *Universiti Putra Malaysia*, Malaysia
ABDUL AZIZ Noor Azeera, *UniversitiTeknologi Malaysia*, Malaysia
ABDULLAH Abdulkareem S., *University of Basrah*, Iraq
ABDULLAH Ahmed Jamal, *Universiti Teknikal Malaysia*, Malaysia
ABU AL-HAIJA Qasem, *Princess Sumaya University for Technology*, Jordan
ABUALIGAH Laith, *Al al-Bayt University*, Jordan
AHMAD Tohari, *Sepuluh Nopember Institute of Technology*, Indonesia
AHMED Saadaldeen Rashid, *Karabük University*, Turkey
AIADI Oussama, *University Kasdi Maerbah Ouargla*, Algeria
AL ETAIWI Wael, *Princess Sumaya University for Technology*, Jordan
ALBAHRANI Ekhlas Abbas, *University of Mustansiriyah*, Iraq
ALI Haider, *Shaanxi Normal University*, China
ALI Mohamed Nabih, *University of Trento*, Italy
ALIABADIAN Ramin, *Islamic Azad University*, Iran
ALIBAKHSHIKENARI Mohammad, *University Charles III of Madrid*, Spain
ALKHATEEB Jawad H., *Prince Mohammad Bin Fahd University*, KSA
ALKHATIB Ahmad, *Al-Zaytoonah University*, Jordan
ALMAHAMID Fadi, *Western University*, Canada
ALTAANI Ahmad T., *Yarmouk University*, Jordan
ALTARAWNEH Mutaz, *Mutah University*, Jordan
ALTUNAY Hakan Can, *Ondokuz Mayıs University*, Turkey
AMAMRA Imed, *University 20 August 1955*, Algeria
AMEUR Yulliwas, *National Conservatory of Arts and Crafts*, France
ASAD Syed Muhammad, *University of Glasgow*, UK
AWOTUNDE Joseph Bamidele, *University of Ilorin*, Nigeria
AYAN Enes, *Kırıkkale University*, Turkey
BACANIN Nebojsa, *Singidunum University*, Serbia
BANDYOPADHYAY Supriyo, *Virginia Commonwealth University*, USA
BARATI Ali, *Islamic Azad University*, Iran
BARATI Hamid, *Islamic Azad University*, Iran
BASHIR Zia, *Quaid-i-Azam University*, Pakistan
BATHLA Gourav, *GLA University Mathura*, India
BEN AYED Yessine, *University of Sfax*, Tunisia
BEN BRAHIM Ghassen, *Prince Mohammad Bin Fahd University*, KSA
BENDJILLALI Ridha Ilyas, *University Center Nour Bachir*, Algeria
BHARDWAJ Aditya, *Bennett University*, India
BHAT Soha Maqbool, *Shri Mata Vaishno Devi University*, India
BOD Mohammad, *Shahid Rajaee Teacher Training University*, Iran
BOUALEM Adda, *University Ibn Khaldoun*, Algeria
BOUCHOUICHA Moez, *University of Toulon*, France
BUTPHENG Chanapha, *National Dong Hwa University*, Taiwan
CAI Haipeng, *Washington State University*, USA
CAO Xiao-Kai, *Sun Yat-sen University*, China
CAO Zhen, *Rice University*, USA
CASTILLO-VALDIVIESO Pedro A., *University of Granada*, Spain
CHATTERJEE Jyotir Moy, *Graphic Era University*, India
CHEN Hao, *National University of Defense Technology*, China
CHEN Hui, *Shanghai University of Electric Power*, China
DARWISH Saad M., *Alexandria University*, Egypt
DAVE Rushit, *Minnesota State University*, USA
DENG Liwei, *Harbin University of Science and Technology*, China
DEZERT Jean, *ONERA*, France
DOAN Thien-Phuc, *Soongsil University*, S. Korea
DOGHMANE Noureddine, *Annaba University*, Algeria
ĐONKO Dženana, *University of Sarajevo*, Bosnia and Herzegovina
DU Jianping, *National Digital Switching System Engineering and Technological Research Center*, China
DUWAIRI Rehab, *Jordan University of Science and Technology*, Jordan
DŽUBUR Benjamin, *University of Ljubljana*, Slovenia
EBERT Nico, *ZHAW University*, Switzerland
ELBOUSHI Ayman, *Electronics Research Institute*, Egypt
EVSUTIN Oleg, *HSE University*, Russia
FAIZ Muhammad Nur, *Politeknik Negeri Cilacap*, Indonesia
FENG Jianwei, *Amazon*, USA
FORBES Andrew, *University of the Witwatersrand*, South Africa
GALINEC Darko, *Zagreb University of Applied Sciences*, Croatia
GAO Xue, *South China University of Technology*, China
GASMI Kaouther, *National Engineering School of Tunis*, Tunisia
GAUR Loveleen, *AMITY University*, India
GAZIS Alexandros, *Democritus University of Thrace*, Greece
GBASHI Ekhlas K., *University of Technology*, Iraq
GEDDAH Hicham, *Mohamed V University*, Morocco
GHARIB Anastassia, *Princess Sumaya University for Technology*, Jordan
GHERABI Noreddine, *University Sultan Moulay Slimane*, Morocco
GIL Sandrine, *University of Poitiers*, France
GRAMMALIDIS Nikos, *Institute of Information Technology and Communications*, Greece
GRITLI Hassene, *University of Tunis El Manar*, Tunisia
HABASH Nizar, *New York University*, USA
HAMMO Bassam, *Princess Sumaya University for Technology*, Jordan
HASHMI Arshad, *King Abdulaziz University*, KSA
HIJJAWI Mohammad, *Applied Science University*, Jordan
HO Sin-Ban, *Multimedia University*, Malaysia
HÖLBL Marko, *University of Maribor*, Slovenia

Name, *Affiliation*, Country

HSU Yung-Lin, *National Taiwan University*, Taiwan
HU Jiangqi, *University of Buffalo*, USA
HUANG Lin-Qing, *Northwestern Polytechnical University*, China
IKRAM Sumaiya Thaseen, *Vellore Institute of Technology*, India
IMOIZE Agbotiname Lucky, *University of Lagos*, Nigeria
JAMIEL Mohamed, *University of Sfax*, Tunisia
JI Jinlong, *Case Western Reserve University*, USA
JIA Ziyu, *Beijing Jiaotong University*, China
JNSSON Arne, *Linkping University*, Sweden
KABIR Muhammad, *Nanjing University of Science and Technology*, China
KANNARI Phanindra Reddy, *National Taiwan University*, Taiwan
KAPOOR Ankush, *Jawaharlal Nehru Government Engineering College*, India
KAYYIDAVAZHIYIL Abhilash, *SSBM*, Switzerland
KAZAR Okba, *Biskra University*, Algeria
KEHINDE Temitope Olubanjo, *Hong Kong Polytechnic University*, Hong Kong
KERSTEN Jens, *German Aerospace Center*, Germany
KHAN Shakir, *Imam Mohammad Ibn Saud Islamic University*, KSA
KHUDAYER Baidaa Hamza, *AlBuraimi University College*, Oman
KIENTOPF Kai, *University Magdeburg*, Germany
KIM Hong-Teuk, *LG Electronics Inc.*, S. Korea
KIM Jaeyun, *Soon Chun Hyanag University*, S. Korea
KOLIANDER Günther, *Austrian Academy of Sciences*, Austria
KRUPALIJA Ehlimana, *University of Sarajevo*, Bosnia and Herzegovina
KUMAR Amit, *IITBHU*, India
KUMAR Mukkesh, *Agency for Science, Technology and Research*, Singapore
LAI WenCheng, *National Taiwan University of Science and Technology*, Taiwan
LATAIFEH Mohammed, *University of Sharjah*, UAE
LEE Bruce W., *Penn Engineering-University of Pennsylvania*, USA
LEI Na, *Dalian University of Technology*, China
LEMEŠ Samir, *University of Zenica*, Bosnia and Herzegovina
LI Junjie, *Wuhan University*, China
LI Qiang, *XATU*, China
LI Wenjuan, *Shanghai Jiao Tong University*, China
LI Wenyu, *Harbin Institute of Technology*, China
LI Yang, *DSP*, China
LI Ye, *WFUST*, China
LI Yi, *Taiyuan University of Technology*, China
LIN Fanzhao, *IIE*, China
LIU Chao, *Civil Aviation University of China*, China
LIU Juntao, *Wuhan Digital Engineering Institute*, China
LIU Jusheng, *Shanghai University of Finance and Economics*, China
LIU Weijian, *Wuhan Electronic Information Institute*, China
LIU Xiaoyuan, *University of Electronic Science and Technology of China*, China
LU Xiaoyan, *Wuhan University*, China
LV Jiming, *NUAA*, China
MADRIA Sanjay, *Missouri University of S&T*, USA
MAFU Mhlambululi, *Case Western Reserve University*, USA
MALLICK Pradeep Kumar, *KIIT University*, India
MARAR Hazem, *Princess Sumaya University for Technology*, Jordan
MEHRIDEHNAVI Alireza, *Isfahan University of Medical Sciences*, Iran
MELANCHTHON Philip, *University of Northwestern*, USA
MELMAN Anna, *HSE University*, Russia
MELNYK Lidiia, *University of Jena*, Germany
MERIT Khaled, *TAHRI Mohamed Bechar University*, Algeria
MOHAMED Nachaat, *Rabdan Academy*, UAE
MOHAMMADI Reza Khan, *University of Guilan*, Iran
MOHANTY Asutosh, *Kalinga Institute of Industrial Technology*, India
MORADI Mohieddin, *IRIB University*, Iran
MUAAD Abdullah Y., *Mysore University*, India
MUSLIM Much Aziz, *Universitas Negeri Semarang*, Indonesia
NAHAR Khalid Mohamed, *Yarmouk University*, Jordan
NGUYEN Tan N., *Sejong University*, S. Korea
NIELSEN Lene, *IT University of Copenhagen*, Denmark
NYATEGA Charles Okanda, *Mbeya University of Science and Technology*, Tanzania
OBEIDAT Mohammad A., *Tafila Technical University*, Jordan
OONISHI Kento, *Mitsubishi Electric Corporation*, Japan
PADHY Neelamadhab, *GIET University*, India
PALMA David, *University of Udine*, Italy
PANAGIOTAKIS Costas, *Hellenic Mediterranean University*, Greece
PENG Ling, *Aerospace Information Research Institute*, China
PINTARD Alice, *Catholic University of Louvain*, Belgium
PRADHAN Nrusingha Charan, *Reykjavik University*, Iceland
RADIUK Pavlo, *Khmelnytskyi National University*, Ukraine
RAHMAN Md Monibor, *Old Dominion University*, USA
RAHMAN Mohammad Masudur, *Dalhousie University*, Canada
RANI Rajneesh, *NITJ*, India
RASTOGI Puru, *Mowito*, India
REN Yanhao, *Fudan University*, China
RIERA Tomás Sureda, *University of Glasgow,* UK
RODRIGUES Carlo Kleber da Silva, *Federal University of ABC*, Brazil
SAAD Saidah, *Universiti Kebangsaan Malaysia*, Malaysia
SAHOO Shreeya Swagatika, *Anusandhan University*, India
SALEEB Demyana Adel, *Kafrelsheikh University*, Egypt

JJCIT Annual List of Reviewers (2023)

Name, *Affiliation*, Country

SAMROUTH Khouloud, *Arab Open University*, Lebanon
SARAEREH Omar, *Hashemite University*, Jordan
SCHÄFER Zeinab, *Aarhus University*, Denmark
SEYEDI Saeid, *National Yunlin University of Science and Technology*, Taiwan
SHAHRIAR Sakib, *American University of Sharjah*, UAE
SHAMSUL Syed Mohammed, *Edith Cowan University*, Australia
SHISHKAREV Vladimir V., *MTUCI*, Russia
SHOJAFAR Mohammad, *University of Surrey*, UK
SHORMAN Samer Mahmoud, *Applied Science University*, Bahrain
SILVA António Rito, *University of Lisbon*, Portugal
SILVA Fernando Selleri, *UNEMAT*, Brazil
SLAMA Rim, *CESI*, France
SOMANTRI Oman, *Politeknik Negeri Cilacap*, Indonesia
SONUÇ Emrullah, *Karabük University*, Turkey
STEINGARTNER William, *Technical University of Košice*, Slovakia
STEWART Kenneth, *University of California, Irvine*, USA
SU Jian, *Nanjing University of Information Science and Technology*, China
SULTANA Jabeen, *Majmaah University*, KSA
SUN Weifang, *Xiamen University*, China
SUN Ying, *Wuhan University of Science and Technology*, China
TAHA Miran, *Valencia Polytechnic University*, Spain
TAHER Fatma, *Zayed University*, UAE
TALAFHA Sameerah, *Southern Illinois University*, USA
TARIQ Muhammad Usman, *Abu Dhabi University*, UAE
TEBACHE Soufiane, *Ecole Nationale Polytechnique*, Algeria
TILAHUN Fitsum Debebe, *Korea University*, S. Korea
TSENG Hsiao-Ting, *National Central University*, Taiwan
ULLAH Zahid, *PAF-IAST*, Pakistan
UPRETI Kamal, *Dr.Akhilesh Das Gupta Institute of Technology and Management*, India
VAHABI Mohsen, *Shahrood University of Technology*, Iran
VARPOSHTI Marzieh, *Amirkabir University of Technology*, Iran
VELAYUTHAM Sivasankaran, *VIT Bhopal University*, India
VERKERKEN Miel, *IMEC-Ghent University*, Belgium
VU Long Nguyen, *Soongsil University*, S. Korea
VYAS Piyush, *Texas A&M University*, USA
WANG Liang, *Xidian University*, China
WANG Weina, *Jilin Institute of Chemical Technology*, China
WATANABE Kazuho, *Toyohashi University of Technology*, Japan
WEI Yuejun, *Huawei Technologies*, China
WILKENS Rodrigo, *Catholic University of Louvain*, Belgium
WINTER Dominique, *HS Emden/Leer*, Germany
WU Yanjie, *University of Electronic Science and technology of China*, China
WU Yuxin, *GZHU*, China
XIE Qi, *Hangzhou Normal University*, China
XU Changqiao, *BUPT*, China
XU Ming, *Hangzhou Dianzi University*, China
YAN Xuesong, China University *of Geosciences*, China
YUAN Jieyu, *Macau University of Science and Technology*, China
ZADA Islam, *Int. Islamic University*, Pakistan
ZHANG Alexander, *Institute of Computer Science*, Greece
ZHANG Changchun, *Beijing Forestry University*, China
ZHANG Hao, *Taiyuan University of Technology*, China
ZHANG Jingyu, *CSUST*, China
ZHANG Na, *West Virginia University*, USA
ZHANG Weidong, *Henan Institute of Science and Technology*, China
ZHANG Weishan, *China University of Petroleum*, China
ZHANG Yi-Xuan, *Xidian University*, China
ZHAO Qinghua, *Nanjing University of Science and Technology*, China
ZHENG Mengce, *Zhejiang Wanli University*, China
ZHOU Nan-Run, *Shanghai University of Engineering Science*, China
ZHOU Zhaoxian, *University of Southern Mississippi*, USA
ZHU Donglin, *Zhejiang Normal University*, China
ZHU Jiaqi, *Harbin Institute of Technology*, China
ZHU Xiaojian, *Changshu Institute of Technology*, China
ZIVKOVIC Miodrag, *Singidunum University*, Serbia

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 السالم، الجبيهة، عمان، الأردن.

صندوق بريد: 1438 عمان 11941 الأردن

هاتف: +962-6-5359949

فاكس: +962-6-7295534

صندوق دعم البحث العلمي والابتكار
Scientific Research and Innovation Support Fund

جامعة الأميرة سميّة للتكنولوجيا
Princess Sumaya University for Technology

المجلة الأردنية للحاسوب وتكنولوجيا المعلومات

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

كانون الأول ٢٠٢٣ المجلد ٩ العدد ٤

JJCIT

www.jjcit.org

jjcit@psut.edu.jo

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