SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V11I1P102

Volume 11 Issue 1, 15-27, January 2024

Original Article

Real-Time Detection and Categorization of Cache Side-Channel Attacks Using Deep Learning and Morlet Wavelet Assistance

C. Lakshminatha Reddy[1], K. Malathi[2]

[1,2]*Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Tamilnadu, India.*

[2]*Corresponding Author : malathi.learning@gmail.com*

Received: 14 October 2023 Revised: 25 November 2023 Accepted: 15 December 2023 Published: 13 January 2024

Abstract - *Cache Side Channel (CSC) attacks track user's cache activity to get private data. This attack often targets the L3 cache that the user and the intruder share. As a result, an intruder can gather private data without tipping off the target. The approach for real-time detection of CSC attacks is developed in this research. The proposed method quickly detects CSC attacks after detecting the change in the CPU numbers. To do this, the value of the CPU counters is measured using a Deep Learning (DL) network with Intel Morlet Wavelet assistance and Weighted Mean of Vectors (WMOV) optimization. Understanding the kind of data that is released is crucial for distinguishing and categorizing cache-based side-channel attacks. Timing attacks, commonly referred to as time-driven attacks, produce quantitative implementation data pertaining to timing. Multiple counters captured changes throughout the attack through the course of the research. The work also presents a categorization of these attacks according to the information-leaking source. After that, a qualitative examination of the effectiveness, complexity, and vulnerabilities of the target cryptosystems of these attacks is conducted. The results of the experiments demonstrate that the proposed detection method performs well for real-time detection in a variety of situations.*

Keywords - *Attack detection, Cache Side Channel, Deep Learning, Intel Morlet Wavelet assistance, Weighted Mean of Vectors (WMOV) optimization.*

1. Introduction

Side-Channel (SC) attacks use unintentional data leaks from computing systems or applications to deduce sensitive data. Numerous monitoring publications examined ways to circumvent cryptographic systems by using timing data, power usage, or Electromagnetic (EM) leakage to steal key information from smart cards [1]. Modelling inverting attacks are the first type, whereby the attacker attempts to retrieve the input provided by the model.

The second kind of attack is model extraction, which is when the attacker attempts to retrieve the victim model's design and/or parameters. The two sorts of attacks constitute a significant threat to the neural network's users or designer, as was previously stated [2]. SC attacks are one of cloud computing's security weaknesses. Concurrent resident Virtual Machines (VMs) shared physical space enables the attacker's and victim's VMs to interact even when they are not intended to do so. Given that cloud computing differs from systems with physical separation [3]. A CPU-side timing channel that controls the timing of GPU operations. They specifically presume that selected text chunks will be encrypted over time using a GPU-based encryption framework. The encryption latency alters based on the encryption key because different memory access patterns cause timing discrepancies owing to memory from the GPU coalescing effects, which allow a timing SC encryption critical attack [4].

A hacker must have a connection to both the target device and the profiling device in order to launch a monitored side-channel attack. The target device that is carrying out an encryption operation with a fixed unknown key value is only partially under the attacker's control. Because attackers can identify the side-channel leaking of the target device before the attack, characterized attacks are regarded as the most potent type of side-channel attacks [5].

This is not only true in a traditional black box environment, though, as every crypto method used on an actual device exposes data about intermediate numbers through SCs like time, power usage, or EM emission [6]. A key component of electronic interaction is coding. The safety of the data handled by sensitive equipment, particularly against so-called SC attacks and fault non-invasive attacks, has recently been demonstrated to be improved by codes.

This study reminds readers that the use of Linear Codes with Complementary Duals (LCD), which linear codes whose crossing with their dual is simple, is crucial for protecting implementations towards these two categories of non-invasive attacks [7]. A previous study on program variety has focused chiefly on randomizing the program participation, such as the stored in memory locations of code and data, since code reuse and associated issues attacks rely on static aspects of a program.

On the opposite hand, SC attacks rely on dynamic program features, such as execution time, memory latencies, or battery life. Therefore, randomizing the carrying out of a program as opposed to its presentation is required for diversity against SCs [8]. All documented auditory SC attacks are passive, which means that the intruder listens in on the target's SC audio signals. The proposed method, however, uses an active SC, which means that the attacker causes the SC's sound emissions [9]. The duration of execution or caching pattern of access of an operation, such as Advanced Encryption Standard (AES) encryption, is often disclosed by SC attacks through a secret channel. Such channels are frequently established in the software that implements encrypted algorithms in a variety of hypervisor-specific procedures and methods, such as storage deduplication, and in the computer's hardware itself, such as CPUs' Last-Level Cache (LLC) [10]. Cache settings may affect how well time-dependent side-channel attacks succeed, according to study findings.

For instance, it would seem logical that a cache with more space would make cyberattacks on AES more challenging since it could store a greater fraction of the precomputed S-box [11]. Although the Intel CPU Optimization Guide lists temporal variances caused by cache-bank disputes, no disclosed attack had ever made use of these vulnerabilities. Intel continues to provide code that employs scatter-gather and suggests using the method for SC prevention despite the lack of a danger that could be shown [12].

1.1. Problem Statement

Traditional methods for detecting CSC attacks often rely on post-mortem analysis or offline monitoring, which can delay the detection and response to an ongoing attack. The research aims to address this limitation by proposing a real-time detection method that can quickly identify CSC attacks as soon as there is a change observed in the CPU numbers. By doing so, the objective is to enable proactive and timely countermeasures to mitigate the potential damage caused by these attacks.

Categorization and Analysis: Another aspect of the problem statement is the need to understand and categorize cache-based SC attacks based on their information-leaking source. This categorization provides insights into the different types of attacks and helps in analyzing their effectiveness, complexity, and vulnerabilities on target cryptosystems. By conducting a qualitative examination of these factors, the research aims to enhance the understanding of CSC attacks and guide the development of appropriate countermeasures.

1.2. Research Contribution

- Detection Method: The research introduces a detection method that utilizes CPU counters and employs DL techniques with Intel Morlet Wavelet assistance and WMOV optimization. This approach enables quick identification of checking for CSC attack changes in CPU numbers.
- By detecting changes in CPU counters, the proposed method enables proactive identification of attacks, reducing the risk of data leakage and allowing for prompt countermeasures to be deployed.
- This categorization aids in understanding the nature of these attacks and facilitates further analysis. Additionally, the research conducts a qualitative examination of the effectiveness, complexity, and vulnerabilities of these attacks on target cryptosystems.
- The research validates the suggested detection technique through experiments conducted in various situations. The results demonstrate the effectiveness of the suggested approach in detecting CSC attacks in real-time scenarios.

The content of this paper is organized in the following order: Section 2 analyzes earlier works on CSC attacks utilizing different optimization models. Section 3 describes the research gap; section 4 briefly explains the proposed method of pre-processing, feature selection and classification. Section 5 is the result and discussion, and finally, section 6 concludes the paper.

2. Related Works

The method introduced by Cho et al. [13] quickly detects attacks through the cache after seeing the change in the CPU numbers. The score of the CPU counters was measured for this using Machine Learning (ML) methods and Intel PCM (Performance Counter Monitor). Multiple PCM counters captured changes along the attack through the course of the study. The findings led to the implementation of a detecting program utilizing these counters. The results of the experiments demonstrated that the suggested detection method performed well for real-time detection in a variety of situations.

Gras et al. [14] suggested ABSynthe, an algorithm that generates additional SCs from inputs of a target program and a microarchitecture. The essential realization was that they regard the target CPU structure as a black box and enable automation by restricting the analysis to (usually on-core) contention-based SCs. The team has automatically produced

leakage maps for several x86_64 microarchitectures to enable ABSynthe. These leakage maps provided a detailed view of the interactions between various x86_64 instructions. They supported the use of a black box methodology to identify the most effective set of instructions for a specific software target, which was also regarded as a black box. ABSynthe used a recurrent neural network to create an effective side attack that retrieve a secret bit stream using the optimized pattern of instructions.

In comparison to contemporary contention-based attacks that concentrate on just one part, the study demonstrated that ABSynthe may synthesize superior attacks by leveraging contention on numerous components at once. In addition, it was possible to synthesize cross-thread attacks for a range of microarchitectures (including Intel, AMD, and ARM) on four distinct cryptographic software targets in both native and virtualized settings due to the mechanization made available by ABSynthe.

Guo et al. [15] proposed the target of the first key-recovery side-channel attack. This innovative approach was to create an attack technique that provided the decoding oracle with unique ciphertexts that matched instances of single mistakes. Such cypher texts could be decoded using just one entry complex secret permutation that was a component of the secret key. One item of the secret permutation may be found by identifying an opening in the additional FFT step that was used to estimate the error locator polynomials. This may be repeated for other lines to get back the entire secret key.

Both the FPGA reference version and a software program running on an ARM Cortex-M4 were used to analyze power to explain the attack. They derive the error locator polynomials from a single trace using a classification approach based on ML. The attack was successfully carried out in practice and was completely integrated and assessed within the Chipwhisperer architecture. In the FPGA scenario, it used fewer than 800 traces for complete key recovery for the lowest parameter set and roughly 300 traces for partial key recovery. An attack against the ARM software implementation required a comparable quantity of traces to be effective.

Shusterman et al. [16] suggested an efficient edge-oriented traffic characterization technique centred around monitoring CPU cache congestion at the last level. This approach conducts direct measures of the user devices using a JavaScript-based API without permission from the website, in contrast with earlier traffic characterization techniques that monitor network traffic from a central point. The analysis demonstrated that the cache-based method's reliability was comparable to network-based approaches' accuracy in both VPN and non-VPN networks. To adequately evaluate the practicality and extendibility of attacks, Batina et al. [17] have conducted all tests on actual data and typical neural net topologies. Real-world outcomes were displayed on an ARM CORTEX-M3 microcontroller. The tests demonstrated that the SC attacker was able to learn about the structures, the neural network's weights, the number of output classes, the number of levels and neurons in each layer, and its activation algorithms. By exploiting SC information, the attacker may successfully hack into the network.

2.1. Research Analysis Summary

The proposed model builds upon existing research, integrating DL, Intel Morlet Wavelet, and WMOV optimization for real-time detection and categorization of CSC attacks. While Cho et al. demonstrated effective detection through changes in CPU numbers, the approach introduced in this model combines DL networks with Morlet Wavelet assistance, aiming to capture nuanced patterns in CPU counter values associated with SC attacks.

Addressing the limitations of prior methods, the model conducts a comprehensive evaluation of adaptability across diverse system architectures and attack scenarios, categorizing attacks based on information-leaking sources. By assessing performance against evolving attack techniques and different cryptographic systems, the proposed model aims to enhance versatility and resilience in CSC attack detection, contributing to the development of a more robust framework for real-world computing environments.

3. Research Gap

While the proposed research makes significant strides in the real-time detection and categorization of cache SC attacks using a combination of DL, Intel Morlet Wavelet, and WMOV optimization, there remains a notable research gap in understanding the adaptability and robustness of the detection method across diverse system architectures and attack scenarios. The study primarily focuses on detecting changes in CPU numbers as an indicator of CSC attacks.

Still, the effectiveness of the approach in scenarios with varying levels of system load, different CPU architectures, or sophisticated attack strategies remains unclear. Additionally, the research emphasizes the categorization of attacks based on information-leaking sources. Still, a comprehensive exploration of the method's performance against evolving attack techniques and its generalizability to different cryptographic systems is lacking. Further investigation was required to validate the suggested detection model's scalability, versatility, and resilience for addressing the dynamic nature of CSC attacks in real-world computing environments.

4. Proposed Model

Analysis from the SC attack, or SC analysis, takes use of implementation-level flaws. In more precise terms, all

calculations carried out on a specific platform cause unintended physiological leakages. These leakages may be thought of as tangible traces left by the device's reaction time, power use, and radiation while it was modifying data. SC attack makes use of the physical signatures with the goal of recovering the key (secret data).

SC attack was first intended to conduct key recovery attacks against encryption technology. SC attack has a few advantages over conventional cryptanalysis, including the ability to use a divide-and-conquer strategy. As a result, an SC attack may be used to recover tiny portions of the key independently rather than testing and recovering the entire key all at once, thus decreasing the level of difficulty of the attack. Figure 1 shows a flow diagram of CSC attacks.

4.1. Data Collection and Pre-Processing

The aim of gathering this dataset is to create SC attacks. In certain positions, with minimal movement and noise (i.e., while a person is standing and sitting), data is gathered from the user. The settings throughout the data collecting procedure are such that what data must be gathered and how frequently must be specified. A consistent instance with a rate of 30 occurrences per second is used to collect the data, as required by references. Whenever a smartphone keyboard is used to type, readings from sensors are produced, and these measurements make up the dataset.

Smartphones come with a variety of sensors. To overcome this constraint, many models have been developed. One model was developed using a dataset of raw acceleration readings, while other models are based on values gathered from other sensors (such as the magnetic and gyroscope). The measurements are kept in a comma-delimited CSV dataset that is organized into 27 keystroke files and timestamped.

4.2. Feature Extraction

The feature extraction process creates a sensor incident window from the data as it is. Each participant's file is chosen to create a window with 500 samples. The chosen window is distinctive enough to allow for the collection of all the readings for the categorization procedure. They give labels since the alphabets are accepted as being true. The design can precisely address and store the sensor readings and utilized alphabets due to this setup. Based on 130,000 sensor raw readings, a feature matrix is created, with every reading consisting of a 3-axis of all three sensors.

4.3. Cache-Based Side-Channels

Attacks present the categorization of cache-based SC attacks in the following paragraphs. Understanding the sort of information that is released is crucial for differentiating and categorizing the attack appropriately. Separates time-driven attacks from trace-driven attacks. Active and passive time-driven attacks are the two subtypes that have been created to categorize time-dependent attacks further.

4.4. Time-Driven Attacks

Timing attacks, commonly referred to as time-driven attacks, produce quantitative operational data pertaining to timing. This kind of time data can communicate covert cryptographic processes. It may be understood as the ratio of cache hits to cache misses while conveying time execution knowledge during encryption. An attacker can obtain the whole key with this kind of time discrepancy. A time-driven attack is depicted in figure, whereby the intruder watches the user's timing data both prior to and following executions.

Therefore, such attacks are effective in estimating the total duration required to recover the victim's confidential data. Time-driven operations may be divided into two groups depending on where the attacker is: active and passive time-driven cache attacks. The victim's computer cannot be accessed by the passive attacker, who is unable to affect it either directly or indirectly. As a result, the attacker is unable to examine the victim's machine's timing data. By executing code on the same workstation, an active attacker can, nonetheless, affect the victim. The attacker may now modify the victim's system since they are adequately informed about the victim's time information. Figure 2 shows a time-driven attack.

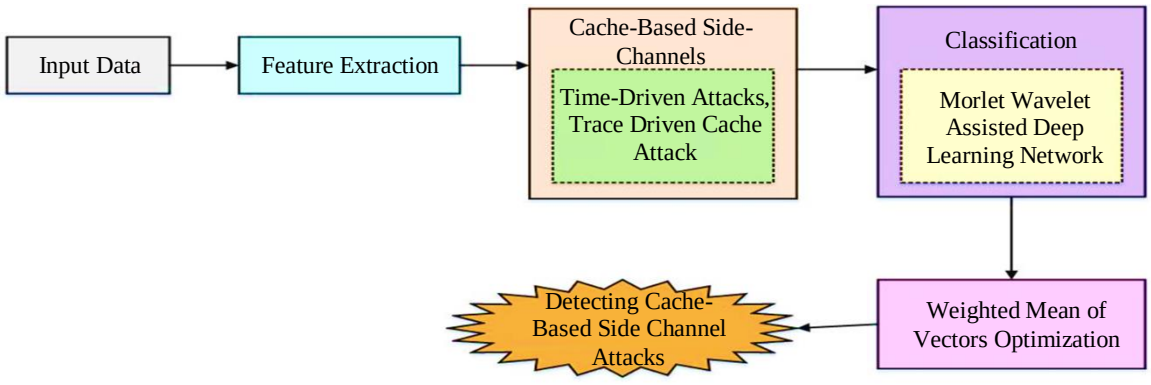

Fig. 1 Flow diagram of proposed model

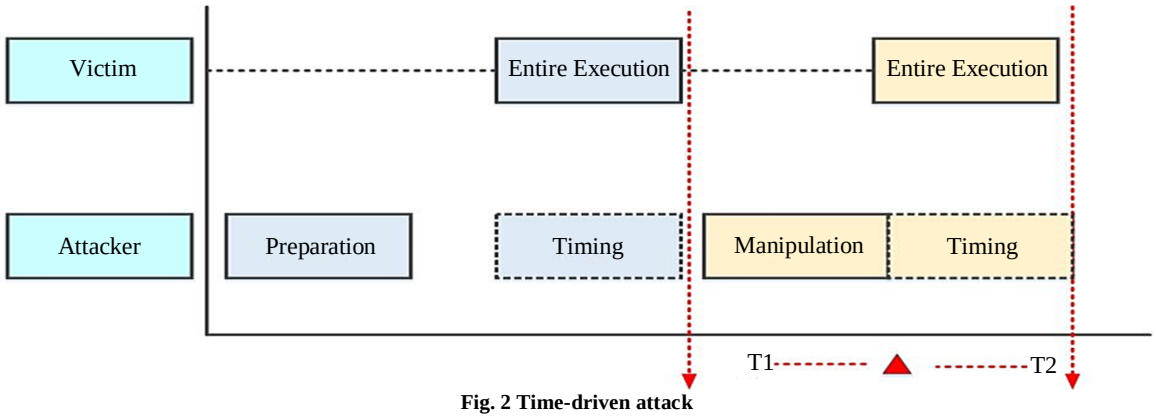

Fig. 2 Time-driven attack

4.5. Trace-Driven Cache Attack

The goal of this attack, as the term implies, is to conceal the victim's access. By continually assessing and analyzing the cache states, as illustrated in Figure 3, this kind of attack seeks to get access to the cache lines that the victim utilized. Access-driven exploits are another name for trace-driven attacks. It is discovered that both time-driven and trace-driven attacks require the victim's system to have been used for operation.

Trail-driven attacks can be more efficient, resourceful, and sophisticated than time-driven attempts if a better trail of data pertaining to the user is gathered. When the hardware supports hyper-threading or Simultaneous Multi-Threading (SMT), which allows for the concurrent running of multiple threads, trace-driven attacks are devastating.

Given that the threads share processing assets, this might be dangerous. Discusses this type of possible attack on RSA where the intruder process may quickly obtain data on squaring and multiplication processes according to the Chinese Remainder Theory (CRT) by watching the L1 activity of RSA encryption. To handle modular functions on private RSA keys, the CRT standard is applied in various RSA implementations. Trace-driven attacks are also found to be more effective and are possible without multiple threads tools. The AES was targeted during the attack, which was conducted on just one threaded machine. Later, a similar approach was carried out to get complete key retrieval for AES encoding, and the attacks have proven to be total functioning asynchronous attacks in the practical setting. A ground-breaking effort that examined the two rounds of AES provided a more quantified study of trace-driven attacks against caches. Attacks that are trace-based also include access-driven attacks. Because they offer detailed information about the victim's ability to access addresses of interest, access-driven attacks are regarded as extremely fine attacks.

4.6. Morlet Wavelet Assisted DL Network Using Classification

This model is a new option for wavelet-based activation functions for neural networks. It maximizes the use of time-frequency localization using the wavelet transform strategy by making use of Morlet wavelets as the activation mechanism for the hidden layer of the basic Autoencoder (AE). Data on nonstationary vibrations are accurately mapped nonlinearly. Since the components of fault characteristic components that are concealed in vibration signals are closer to their activation function. Morlet wavelet transmission is given as follows:

$$\psi(h) = \frac{1}{\sqrt{f_b}} cos(2\pi f_c h)\, exp(-h/f_b) \qquad (1)$$

A Morlet wavelet function is optimized by f_b, f_c; f_c and f_b are the central frequency and bandwidth, respectively. Figure 4 shows conv and Morlet wavelet DL architecture.

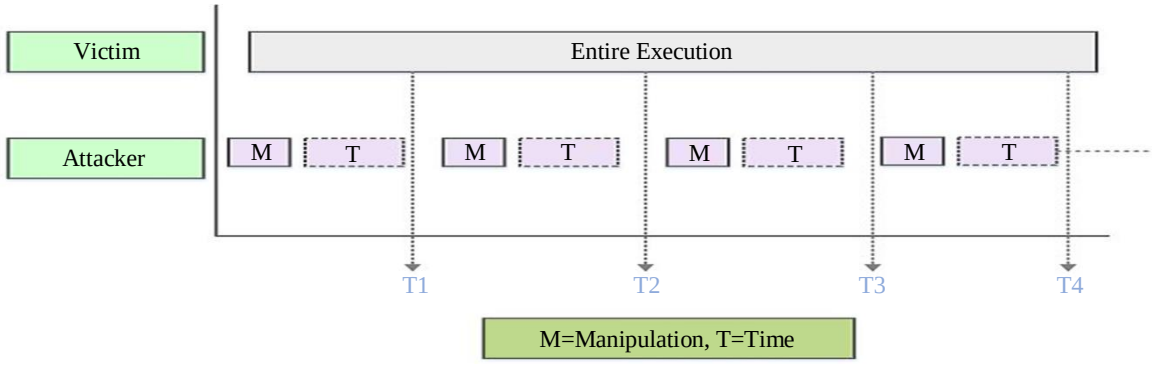

Fig. 3 Trace-driven attack

Here, the sample of input $d = [d_1, d_2, \ldots, d_n]$, then the output of the hidden layer is given by,

$$t_j = \frac{1}{\sqrt{f_b \pi}} cos\left(2\pi f_c \left(\sum_{k=1}^{m} C_{jk} d_k - w_j\right)/x_j\right) . exp\left(-\left(\left(\sum_{k=1}^{m} C_{jk} d_k - w_j\right)/x_j\right)^2 / f_b\right) \quad (2)$$

Here t_j denotes the hidden layer's output j, x_j and w_j are the shift factor and scale factor; correspondingly, hidden node weight is C_{jk} of j and node of input k, C_{ij} is the hidden node weight j over output node i. The $tan\,h$ output is by setting the nonlinear transformation. A Modified AE model (MAE) is constructed using the Morlet wavelet activation function, as shown in Figure 5. The fitness function f indicates the predicted pattern's function p. The below equation gives it:

$$f(p) = \frac{(H(t,p) - H(t))}{H(t)} \quad (3)$$

In cross-entropy $(H(t,p))$ approaches, the entropy of $t(H(t))$, p is a better estimate for t. When $t = p$, the cross entropy is just the entropy of t, and the value of the fitness function becomes zero, which is a minimal value.

4.7. Weighted Mean of Vectors Optimization

A new WMOV optimizer will be used to derive the architectural parameters (for instance, weight) of the Morlet wavelet. Here, the weighted average of the vector optimization approach is used to achieve high accuracy. In this paper, the optimization algorithm is introduced. Below are thorough explanations of a specific place inside a system or item, a process, and ideas. The WMOV is used to determine the weighted mean in the search space. The suggested method demonstrates that a workable solution entails a collection of vectors that are determined by the weight population. The vector's points are updated using one of three operations:

1. Updating the rule
2. Combining vector
3. Using local search

4.7.1. Initialization

WMOV is composed of a total number of weight (Np), weight population in the dimensional of D the search domain $(Y_{l,j}^{g} = \{y_{l,1}^{g}, y_{1,2}^{g}, \ldots y_{1,D}^{g}\}, l = 1,2,\ldots,Np)$. Some control settings are provided, along with a description of this point for the WMOV algorithm. The two key variables are the weighted mean and the scaling factor. Boost the resulting vector using the new rule. When employing the scaling rate, operators must be dependent on the search domain. This σ factor is used to weigh the mean scale of the vector. Its value is calculated using a formula based on a feasible problem, an exponential function and search space. The user is not required to change the generation in accordance with the two parameters.

4.7.2. Updating the Rule

The weight population's update rule operator employs the WMOV algorithm during the search process. The operator's WMOV is used to construct additional vectors. Similar to the WMOV method, additional algorithms are distinguishable from it by this operator and have two primary components. In the first section, an average-based algorithm is used to extract a weighted mean from the collection of random vectors. A mean-based approach initiates with a random starting key and progresses to the next solution based on randomly chosen vectors using weighted average information. It speeds up convergence and improves the algorithm's effectiveness in finding the best solutions.

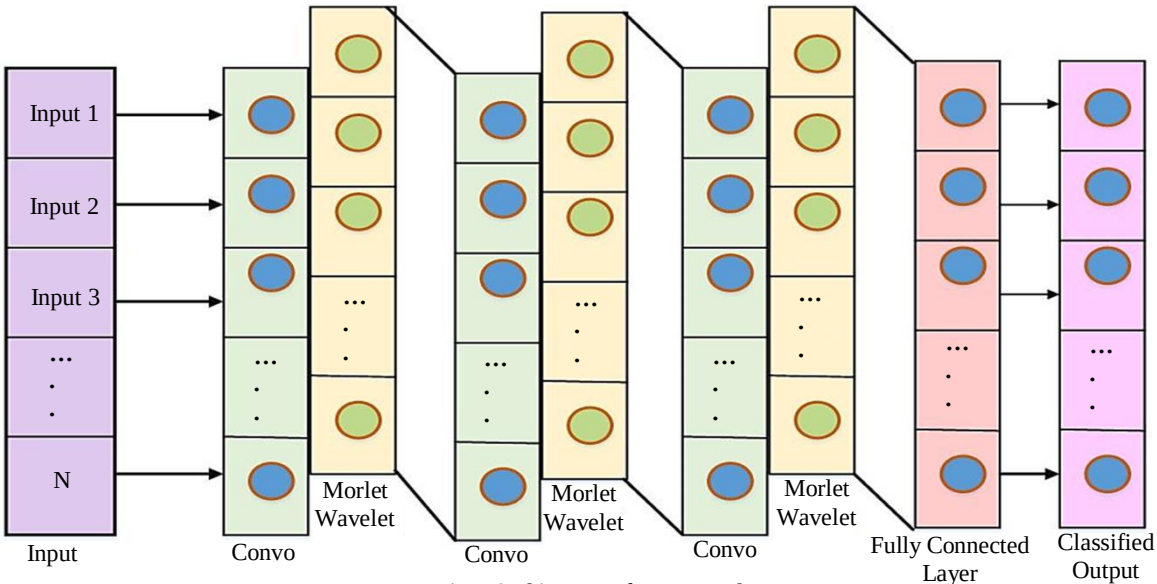

Fig. 4 Architecture of DL network

Instead of changing the current vector, WMOV employed a series of differential vectors that were randomly picked to achieve the weighted mean. For this work, the best is $MeanRule$ by population's diversity increasing. The top five solutions (objective function) are randomly determined to be the best solutions. Likewise, the mean-based rules are applied to the $MeanRule$, as shown in Equation (4):

$$MeanRule = j \times WM1_l^g + WM2_1^g \quad l = 1,2,\ldots,NP \tag{4}$$

$$WM1_l^g = \delta \times \frac{c_1(d_{a1}-d_{a2})+c_2(d_{a1}-d_{a3})+c_3(d_{a2}-d_{a3})}{c_1+c_2+c_3+\varepsilon} + \varepsilon \times rand, l = 1,2,\ldots..,NP \tag{5}$$

Here,

$$c_1 = cos(\,(f(d_{a1}) - f(d_{a2})) + \pi) \times exp(-\frac{f(d_{a1})-f(d_{a2})}{\omega}) \tag{6}$$

$$c_2 = cos(\,(f(d_{a1}) - f(d_{a3})) + \pi) \times exp(-\frac{f(d_{a1})-f(d_{a2})}{\omega}) \tag{7}$$

$$c_3 = cos(\,(f(d_{a2}) - f(d_{a3})) + \pi) \times exp(-\frac{f(d_{a2})-f(d_{a3})}{\omega}) \tag{8}$$

$$\omega = max(f(d_{a1}), f(d_{a2}), f(d_{a3})) \tag{9}$$

$$WM2_l^g = \delta \times \frac{c_1(d_{bs}-d_{bt})+c_2(d_{bs}-d_{ws})+c_3(d_{bt}-d_{ws})}{c_1+c_2+c_3+\varepsilon} + \varepsilon \times rand \;\; l = 1,2,\ldots.,Np \tag{10}$$

Here, $f(d)$ denotes objective function; $z1 \neq z2 \neq z3 \neq l$ are the range of $[1, NP]$ are different integers randomly selected; ε denotes the small value of a constant number; $randn$ indicates random value normally distributed; d_{bs}, d_{bt}, and d_{ws} are better, best, and bad solutions, along with all vectors in weight population meant g^{th} accordingly, generational. After sorting the solution, iterations are determined by this solution. r indicates a random number range of [0, 0.5]; the proposed WMOV algorithm searches globally for solutions by using the weighted means vectors c_1 , c_2, and c_3. During the optimization process, dilation parameters were varied by employing equations. In equation (15), δ denotes the scale factor, and β the function may change on an exponential.

$$\delta = 2\beta \times rand - \beta \tag{11}$$

$$\beta = 2\,exp(-4 \times \frac{g}{max\,g}) \tag{12}$$

Here, $max\,g$ denotes the maximum number of generations. The best vector is now being used in the search space, and the CA (Convergence Acceleration) component adds a rule update operator to provide the ability to perform a global search. The WMOV algorithm supports the crossest solution to the global optimum as the best option.

$$CA = randn \times \frac{(d_{bs}-d_{z1})}{(f(d_{bd})-f(d_{z1})+\varepsilon)} \tag{13}$$

The random number range is [0, 1] here $randn$ denotes a random number accompanied by a normal distribution. To end with, using the equation, a new vector is computed:

$$b_l^g = d_l^g + \sigma \times MeanRule + CA \tag{14}$$

To discover a promising search domain space (exploration phase), it should generally search globally using an optimization algorithm. Similarly, the suggested updating rule based on d_{bs}, d_{bt}, ${d_l}^g$, and ${d^g}_{a1}$ is using the following scheme, $rand$ is less than 0.5:

$$z1_l^g = d_l^g + \sigma \times MealRule + randn \times \frac{(d_{bs}-d_{a1}^g)}{(f(d_{bs})-f(d_{a1}^g)+1)} \tag{15}$$

$$z2_l^g = d_{bs} + \sigma \times MeanRule + randn\,\frac{(d_{a1}^g-d_{a2}^g)}{(f(d_{a1}^g)-f(d_{a2}^g)+1)} \tag{16}$$

Here, $z1_l^g$ and $z2_l^g$ are g^{th} a group of new vectors; and σ denotes a vector's velocity of scale, as the defined equation. As seen in the equation, α, the exponential function defining the function can be modified.

$$\sigma = 2\alpha \times rand - \alpha \tag{17}$$

$$\alpha = w\,exp(-x \times \frac{g}{max\,g}) \tag{18}$$

Here, w and x are constant numbers, correspondingly. When the parameter is large, it is worth noting that σ, the recent position, advances towards the exploitation search if this parameter is lowered. Still, the present location moves towards the WMOV (exploitation search) when this parameter is increased.

4.7.3. Combining Vector

In order to enhance the population's diversity in this study, $z1_l^g$ and $z2_l^g$ are two vectors intended in the earlier section $rand < 0.5$ to create a new vector e_l^g. This operator provides a new and promising way to promising way to promote local search:

$$e_l^g = \begin{cases} z1_l^g + \mu.\left|z1_l^g - z2_l^g\right| & rand < 0.5 \\ 1 & otherwise \end{cases} \tag{19}$$

$$e_l^g = d_l^g \tag{20}$$

Here, e_l^g is obtaining vectors by the vector combining in g^{th} generation; $0.05 \times rand$ is equal to μ.

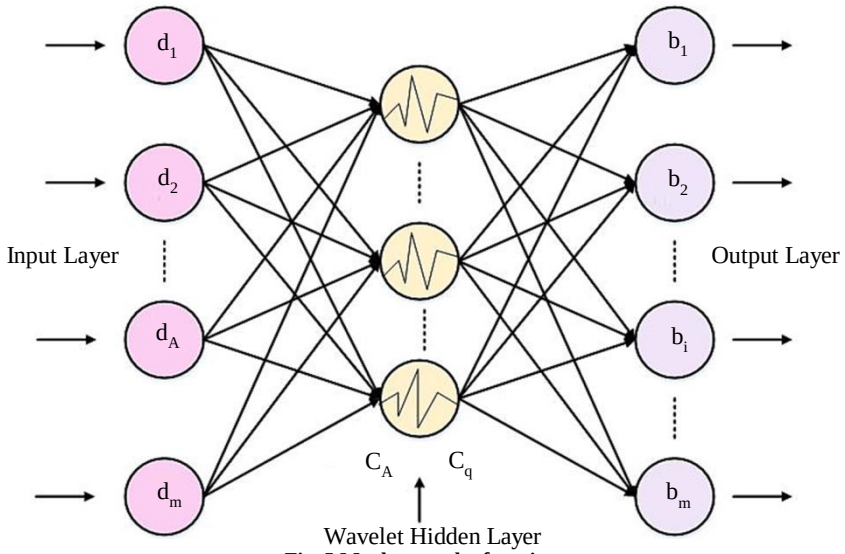

Fig. 5 Morlet wavelet function

4.7.4. Using Local Search

By utilizing efficient local search capabilities, WMOV may be protected from deceit and falling into locally optimum solutions. By utilizing the global location data, it is believed that the local operator may encourage exploitation, convergence, and search for the global optimum (d_{best}^{g}) to guarantee further that the global optimal location is in the best position.

The mean-based rule stated in the equation is also used. If $r < 0.5$, where $rand$ is a random value in [0, 1], a new vector can be generated around d_{best}^{g}:

$$e_{l}^{g} = \begin{cases} d_{bs} + randn \times \left(MeanRule + randn \times \left(d_{bs}^{g} - d_{a1}^{g}\right)\right) & if\ rand < 0.5 \\ 1 & otherwise \end{cases} \quad (21)$$

In which,

$$d_{rnd} = \varphi \times d_{avg} + (1 - \varphi) \times (\varphi \times d_{bt} + (1 - \varphi) \times d_{bd} \quad (22)$$

$$d_{avg} = \frac{(d_a + d_b + d_3)}{3} \quad (23)$$

Here, φ indicates a random number range of (0,1); and d_{rnd} combines $d_{avg,}$ d_{bt}, and d_{bs} randomly. The solution space of the suggested algorithm is better searched by increased randomness nature. The two random numbers υ_1 and υ_2 defined as:

$$\upsilon_1 = \begin{cases} 2 \times rand & if\ l > 0.5 \\ 1 & otherwise \end{cases} \quad (24)$$

$$\upsilon_2 = \begin{cases} rand & if\ l < 0.5 \\ 1 & otherwise \end{cases} \quad (25)$$

In this case, l is a random number (0 and 1). If the random number υ_1 and υ_2 are added to the vector, the best position impact may be increased. The total vectors according to the computational complexities of WMOV. The number of objects and total quantity of iterations are designed as follows:

$$O(CMV) = O(G \times (N \times d)) = O(GNd) \quad (26)$$

Here, N indicates the number of vectors (weight populace), G denotes supreme generation, and d indicates the number of objects.

5. Results and Discussion

This section includes descriptions of the experiment setup, performance measurement, dataset for evaluation, and results. The research method is developed using the Python platform, and the experimental outcomes are assessed and compared with prior ML prediction models in terms of statistical metrics like CEP, F1-Score, and MCC. The tests are conducted using an Intel Corei5-3461 CPU, which has four logical processors and cores and operates at 3.20GHz. The system has an internal physical memory (RAM) of 8GB.

Figures 6 and 7 show the accuracy and loss values as they indicate whether it performs after each optimization cycle. The performance of the research model was computed utilizing the accuracy parameter. A model's accuracy was often computed after the model's input parameters and was expressed as percentages.

The degree to which a measurement is accurate in relation to its actual value. The extent to which repeated measurements produce identical results under the same circumstances is known as precision. Compared to the Naive Bayes (NB), Random Forest (RF), and Decision Trees (DT), the proposed method gives the best result. The proposed method gives the best accuracy value (98.4).

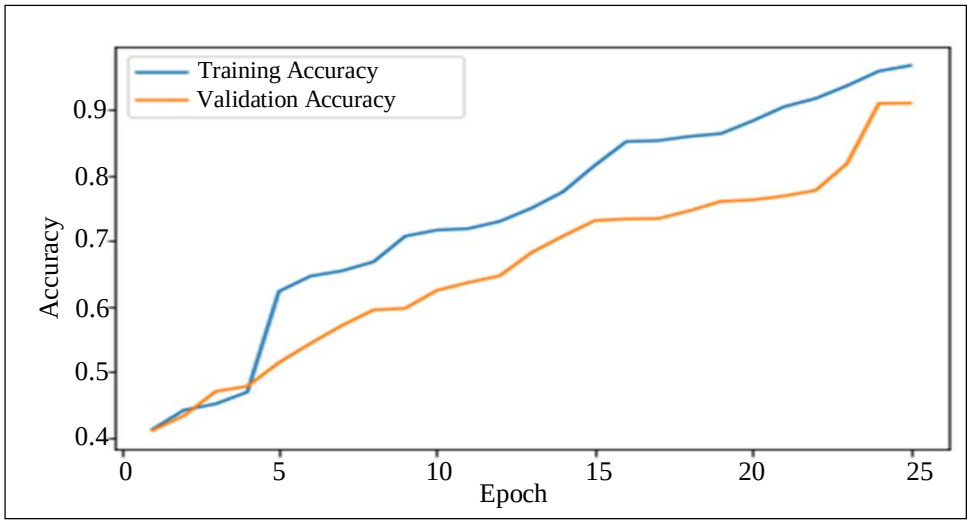

Fig. 6 Accuracy vs. Epoch comparison

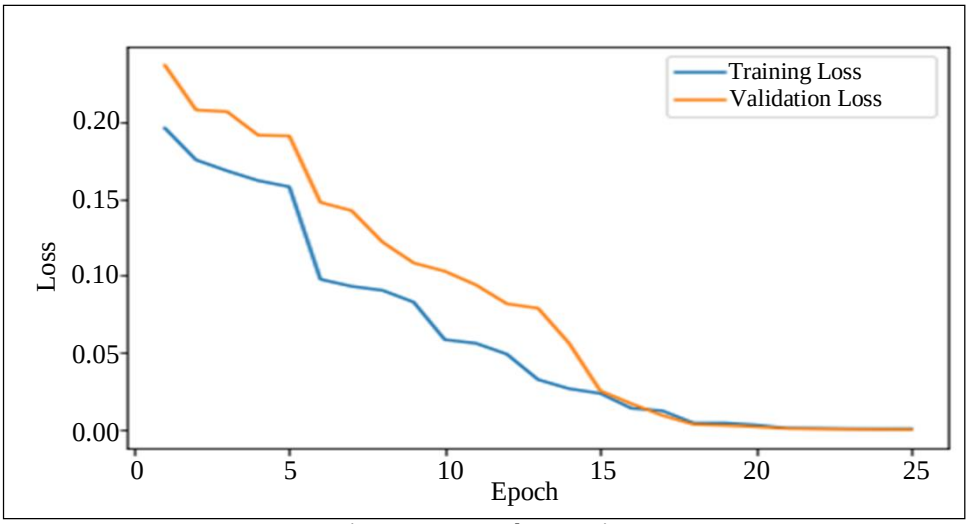

Fig. 7 Loss vs. Epoch comparison

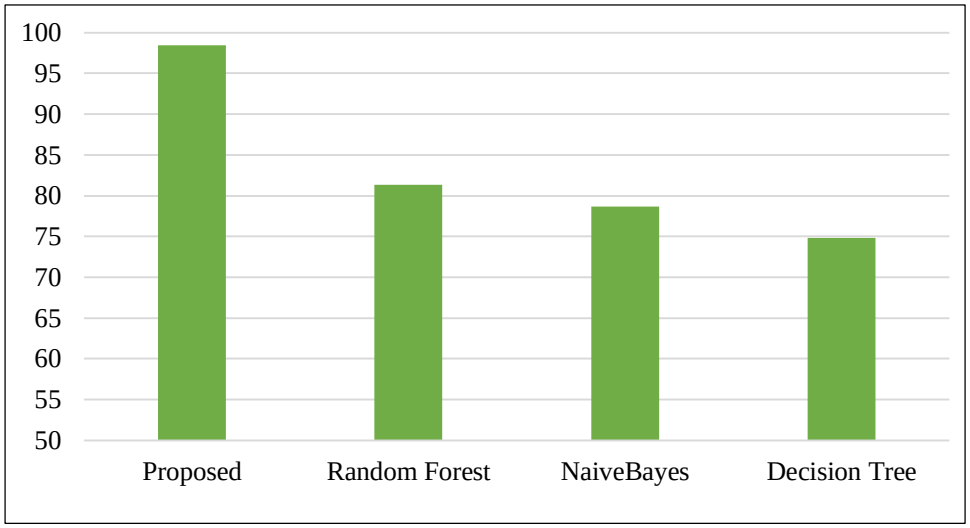

Fig. 8 Accuracy comparison

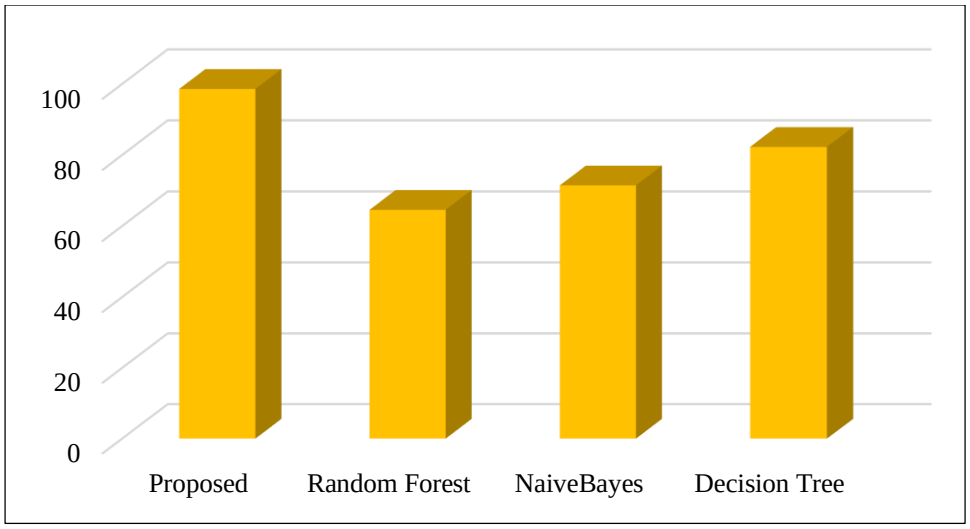

Fig. 9 Precision comparison

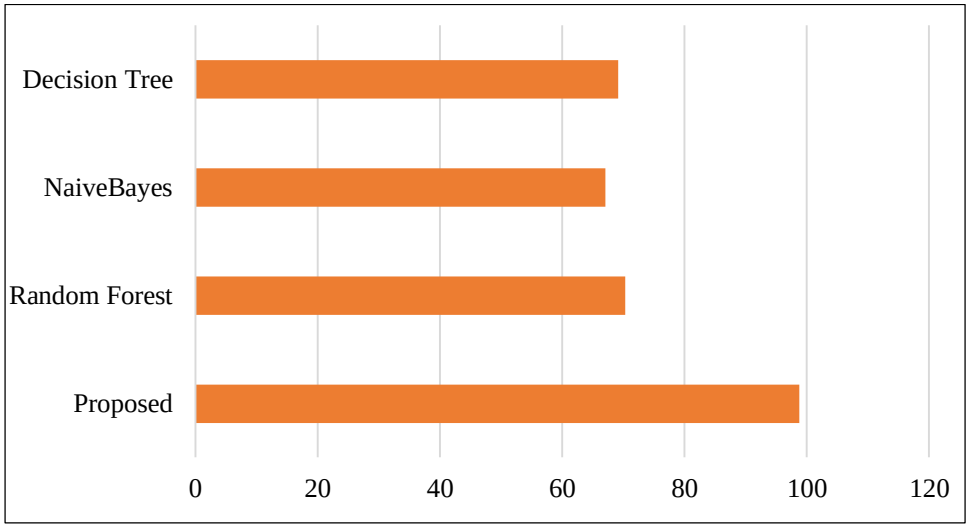

Fig. 10 F-measure comparison

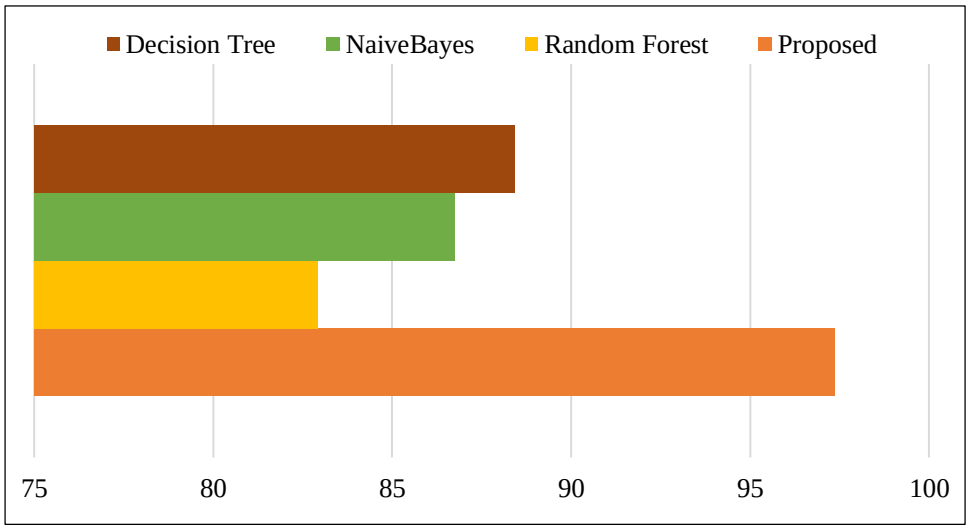

Fig. 11 Specificity comparison

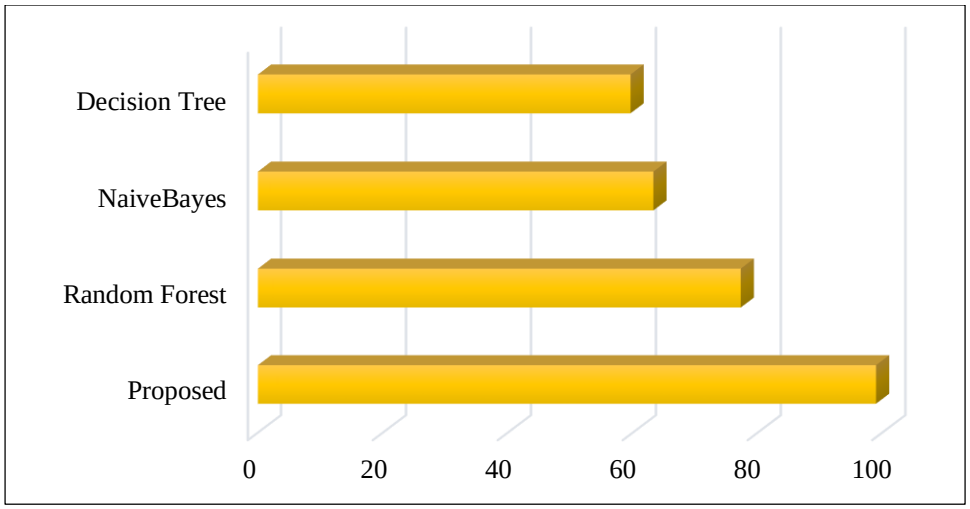

Fig. 12 Confusion matrix of ORSUS-BDT algorithm

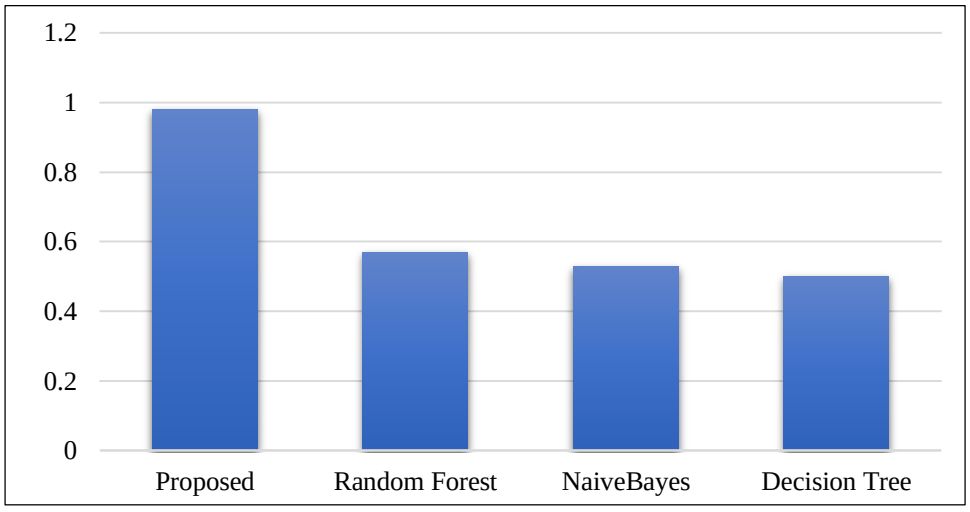

Fig. 13 MCC comparison

Figure 9 illustrates the connection between recall and precision (also known as positive predictive value) and the percentage of pertinent occurrences that were retrieved. Recall is also referred to as sensitivity. Compared to the Random Forest, Naive Bayes, and Decision Tree, the proposed method gives the best result.

The F-measure is calculated using the harmonic median of precision and recall, with equal weights given to either. This allows for the accounting of both accuracy and recall in a single score, allowing for the comparison of models and describing an algorithm's efficacy. Figure 10 compares the Random Forest, Naive Bayes, and Decision Tree; the proposed method gives the best result.

Figure 11 illustrates specificity-the ratio of true negatives that the method correctly identifies-as a percentage. It means that additional real negatives, called false positives since they are previously believed to be positives, would be reported. A True Negative Rate (TNR) might additionally be utilized to describe this percentage. Compared to the RF, NB, and DT, the proposed method gives the best specificity result.

The framework's hypotheses are ranked on the sensitivity chart from most critical to least essential, as shown in Figure 12. A presumption has a considerable impact on the prediction if there is a high relationship between the projection and the presumption. Compared to the RF, NB, and DT, the proposed method gives the best sensitivity result.

MCC was frequently utilized and highly considered as the balanced parameter ideal for classification with imbalance concerns in cache-based side-channel attacks.

$$\mathrm{MCC} = \frac{(TP*TN-FP*FN)}{\sqrt{(TP+FN)(TP+FP)(TN+FP)(TN+FN)}} \qquad (27)$$

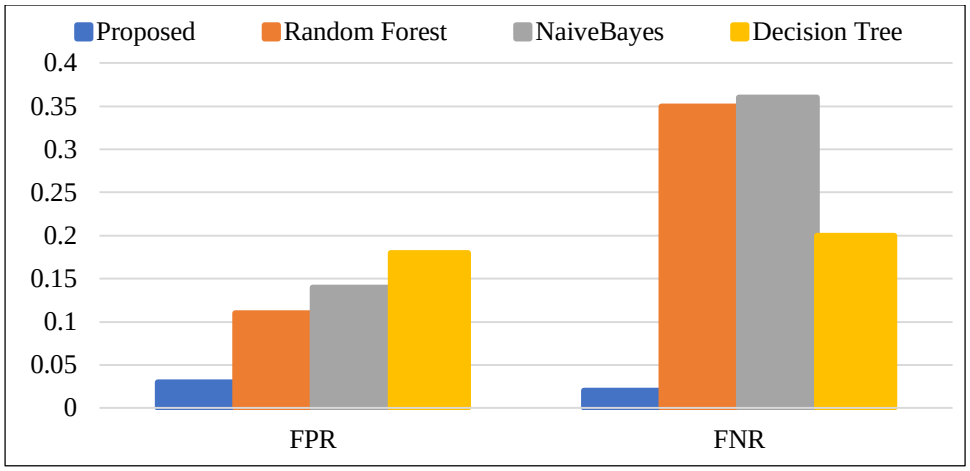

Fig. 14 Error metrics comparison

Instead, if the prediction is accurate, MCC generates a maximum score, which is more dependable. Figure 13 compares the MCC with the suggested way, and the research model provides better results. The FNR and FPR are the error parameters. Figure 14 shows the comparison of the error parameters with other models.

$$FNR = \frac{FN}{FN+TP} = 1 - TPR \quad (28)$$

$$FPR = \frac{FP}{FP+FN} = 1 - TNR \quad (29)$$

The FPR for the proposed technique was 0.02, and the FNR was 0.26; the FPR for the RF model was 0.17, and the FNR was 0.22; the FPR for the NB method was 0.13, and the FNR was 0.36; and the FPR for the DT method was 0.11, and the FNR was 0.40. The suggested method's error metrics are small when compared to those of existing techniques.

5.1. Advantages of Proposed Model

The proposed model offers several advantages in the domain of CSC attack detection. By integrating DL, Intel Morlet Wavelet, and WMOV optimization, the model enhanced the precision and efficiency of real-time detection, surpassing prior methods that rely solely on changes in CPU numbers. The incorporation of Morlet Wavelet provides a sophisticated means to capture nuanced patterns in CPU counter values associated with SC attacks, allowing for more accurate identification. Additionally, the model addresses the need for adaptability and robustness by conducting a thorough evaluation across diverse system architectures and attack scenarios, contributing to a versatile and resilient detection framework. The categorization of attacks based on information-leaking sources further enhances the model's utility, providing a comprehensive approach to understanding and mitigating different types of CSC attacks.

5.2. Limitations of Proposed Model

Despite its strengths, the proposed model has certain limitations. One potential challenge lies in the complexity introduced by the integration of DL and wavelet techniques, which may require substantial computational resources and specialized knowledge for implementation. The model's performance may be influenced by the quality and quantity of training data, and its effectiveness across all possible attack scenarios remains to be validated.

Additionally, the generalizability of the proposed approach to diverse cryptographic systems and real-world computing environments warrants careful consideration, as the model's efficacy may vary depending on specific system configurations. Ongoing research and practical implementation efforts are necessary to address these limitations and ensure the model's reliability and applicability in a broader range of contexts.

6. Conclusion

In conclusion, the proposed method for real-time detection of CSC attacks utilizing CPU counters, DL with Intel Morlet Wavelet assistance, and WMOV optimization proves to be effective in detecting such attacks in a timely manner. By monitoring changes in the CPU counters, this approach can quickly identify CSC attacks without raising suspicion from the target. The research also highlights the importance of understanding the nature of data leakage in cache-based SC attacks. Categorizing these attacks based on their information-leaking source makes it easier to analyze their effectiveness, complexity, and vulnerabilities on target cryptosystems. Further research can explore enhancing the detection method to cover a broader range of CSC attacks. This could involve investigating new features or metrics that capture different aspects of cache activity to improve accuracy and robustness.

Acknowledgements

The authors would like to thank the Saveetha School of Engineering and Saveetha Institute of Medical and Technical Sciences, Chennai, Tamilnadu, India, for their support and motivation throughout this research.

References

[1] Raphael Spreitzer et al., "Systematic Classification of Side-Channel Attacks: A Case Study for Mobile Devices," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 465-488, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[2] Hervé Chabanne et al., "Side Channel Attacks for Architecture Extraction of Neural Networks," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 3-16, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[3] Shahid Anwar et al., "Cross-VM Cache-Based Side Channel Attacks and Proposed Prevention Mechanisms: A Survey," *Journal of Network and Computer Applications*, vol. 93, pp. 259-279, 2017. [CrossRef] [Publisher Link]

[4] Hoda Naghibijouybari et al., "Rendered Insecure: GPU Side Channel Attacks are Practical," *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2139-2153, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[5] Benjamin Timon, "Non-Profiled Deep Learning-Based Side-Channel Attacks with Sensitivity Analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 2, pp. 107-131, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[6] Prasanna Ravi et al., "Generic Side-channel Attacks on CCA-Secured Lattice-Based PKE and KEMs," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 3, pp. 307-335, 2020. [CrossRef] [Publisher Link]

[7] Claude Carlet, and Sylvain Guilley, "Complementary Dual Codes for Counter Measures to Side-Channel Attacks," *Coding Theory and Applications*, pp. 97-105, 2015. [CrossRef] [Publisher Link]

[8] Stephen Crane et al., "Thwarting Cache Side-Channel Attacks through Dynamic Software Diversity," *Proceedings 2015 Network and Distributed System Security Symposiums*, San Diego, pp. 1-14, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[9] Peng Cheng et al., "SonarSnoop: Active Acoustic Side-Channel Attacks," *International Journal of Information Security*, vol. 19, pp. 213-228, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[10] Xiaodong Yu et al., "Comparative Measurement of Cache Configurations' Impacts on Cache Timing Side-Channel Attacks," *12th USENIX Workshop on Cyber Security Experimentation and Test (CSET 19)*, pp. 1-9, 2019. [Google Scholar] [Publisher Link]

[11] Yuval Yarom, Daniel Genkin, and Nadia Heninger, "CacheBleed: A Timing Attack on OpenSSL Constant-Time RSA," *Journal of Cryptographic Engineering*, vol. 7, pp. 99-112, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[12] Jonghyeon Cho et al., "Real-Time Detection for Cache Side Channel Attack Using Performance Counter Monitor," *Applied Sciences*, vol. 10, no. 3, pp. 1-14, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Ben Gras et al., "ABSynthe: Automatic Blackbox Sides-Channel Synthesis on Commodity Microarchitectures," *Network and Distributed Systems Security (NDSS) Symposiums*, San Diego, USA, pp. 1-18, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[14] Qian Guo, Andreas Johansson, and Thomas Johansson, "A Key-Recovery Side-Channel Attack on Classic McEliece Implementations," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2022, no. 4, pp. 800-827, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[15] Anatoly Shusterman et al., "Cache-Based Characterization: A Low-Infrastructure, Distributed Alternative to Network-Based Traffic and Application Characterization," *Computer Networks*, vol. 200, 2021. [CrossRef] [Publisher Link]

[16] Lejla Batina et al., "CSI Neural Network: Using Side-Channels to Recover Your Artificial Neural Network Information," *arXiv*, pp. 1-15, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[17] Maria Mushtaq et al., "Winter is Here! A Decade of Cache-Based Side-Channel Attacks, Detection & Mitigation for RSA," *Information Systems*, vol. 92, 2020. [CrossRef] [Google Scholar] [Publisher Link]