Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol. 14, Issue 5 (2025)
https://www.jneonatalsurg.com

Multi-Domain Steganalysis Preprocessing to Fusion Feature for Optimal Stack Ensemble Model

Malige Gangappa[*1], Balla V V Satyanarayana[2]

[*1]Department of CSE,VNR VJIET, Hyderabad, Telangana, India -500090.

Email ID: gangappa_m@vnrvjiet.in

[2]Department of CSE,VNR VJIET, Hyderabad, Telangana, India -500090.

Email ID: 23071d5801@vnrvjiet.in

Cite this paper as: Malige Gangappa, Balla V V Satyanarayana, (2025) Multi-Domain Steganalysis Preprocessing to Fusion Feature for Optimal Stack Ensemble Model. *Journal of Neonatal Surgery*, 14 (5), 275-295.

ABSTRACT

The field of image-based steganography has been widely used, because of the advancement of steganography methods and their applications. In today's world, image-based exploits are used by the steganography approaches in the publicly available dataset. This dataset is used for data modeling to tune the model for high accuracy, robustness, and other best-fit parameters. So, this paper aims to introduce a novel way of approaching hybrid-based steganalysis, including two algorithm blocks. The first block consists of JPEG-based pre-processing as an initial-level stego cross-verification match using multi-domain steganalysis such as statistical, structural, and frequency. The second block consists of a custom-based fusion feature extraction and meta-feature analysis stage based on the statistical measure evaluation with the machine-level models and their stacked ensemble classification, which improved the analysis of the stego and cover images. As a result, our approach would be lightweight for integration modules for different areas like the initial level for data security to minimize individual and organizational hardware-level stego JPEG-image-based exploits with exception flow management, our model will enhance computational efficiency and higher performance scores of steganalysis.

Keywords: *multi-domain steganalysis, fusion feature extraction, hybrid optimization search-classifier, optimal stacking ensemble model.*

1. INTRODUCTION

Today's world widely uses the internet to communicate from one point to another point for faster transmission of information. By using an application, internet communication is established between the sender, receiver, server, and client. That application supports different types of multimedia like text messages, audio, images, videos, etc. As part of the security of data transmission over the network, nodes must be included to avoid data leaks and public visibility. The researcher developed different methods like watermarking, cryptography, and steganography, which are related and have different purposes of usage in real-time communication.

Watermarking is the process of embedding external information like textual, metadata, or logos into the multimedia of images, videos, audio, or documents. Here, the main purpose of watermarking is to protect the user's uniqueness by offering the ownership of assert, intellectual property rights, or tracking their media over the medium. The watermarking can be applied visibly or invisibly, which does not affect the visual or auditory experience of the main image. This can be used for content authenticity. Cryptography is used for securing communication over the network using the different mechanisms types. It involves techniques like encryption-decryption, digital signatures, and hashing. Here the data will be transformed from normal data to an unrecognizable form with a key. This can provide the privacy of users, data integrity, and authenticity.

Whereas, steganography is the art and science of hiding the secret information within a multimedia such as images, audio, or video file. The main goal of steganography is to hide data in media, and in case the transmission is leaked by the attacker, secret data will be hidden and unknown to them. Unlike watermarking, which may be visible or invisible based on the user choice, it can easily be detectable as compared to steganography, and also data capacity is lower. On the other hand, compared to cryptography, it focuses on making data unreadable to the attacker and unauthorized users over the communication channel, whereas steganography is lightweight and focuses more on the attacker's ability to detect the hidden data pattern techniques used.

In the modern world, looking forward to improving the data modeling using various datasets available publicly. This can be the chance for the hackers to take over individual and small capital organizations over their control by using the advantages of steganography techniques to inject malicious secret code into the dataset images. Because these tampered datasets are undergoing different stages and deployed on the user type of local system level and production server to fine-tune the automation model causes huge damage and losses. Therefore, in this paper, we propose an approach of a two-stage blocks integrated algorithm that combines cross-verification of stego and cover jpeg images using multi-domain correlation and improvised stacked ensemble classification with fusion feature extraction for improved steganalysis. The total number of important feature selections is designed to reduce the computation and optimize resource management for the stack model. We also constructed the meta-model learner component to reduce the dimensionality and prevent overfitting in the model.

The main contributions of our proposed approach are as follows below:

1) The initial block approach is designed based on the multi-domain of steganalysis to cross-verification for preprocessing the jpeg image into simulated folders as proposed.
2) The fusion feature extraction is implemented for the initial block's output to train-test several base classifiers and stack ensembles to further compare their accuracy.
3) We constructed a meta-model using four components that can be used for the important feature selection to prevent excess training and improve the overall model stability.
    - The rest of this paper is organized as follows. In Section II, the related research works of steganography and steganalysis. In Section III, The proposed approach of steganalysis two-stage block is described. In section IV, implementation of the proposed approaches algorithm. In Section V, experimental analysis and their results. In section VI, we give details about the conclusion and future work of this research paper directions.

2. RELATED WORKS

This section provides an overview of steganography and steganalysis works done by various research authors in information security.

The origin of the steganography process existed in an ancient place called the land of the Greeks, as evidenced by Herodotus' stories. In one of the cases, Histiaeus tattooed a message to Aristagoras on the shaved head of a slave and waited for the hair to grow back before sending him to deliver it. Demaratus warned Greece of an impending assault by writing on the wooden back of a wax tablet before adding the beeswax surface. Aeneas Tactician recommended utilizing women's jewelry to conceal secret information and pigeons to transmit secret communications.

Johannes Trithemius used the term "steganography" for the first time in early writing in 1499 when he wrote Steganographia, a tract on cryptography and steganography that was passed off as a book about magic [18]. Additionally, Trithemius created the "Ave-Maria-Cipher" in his work Polygraphiae, which conceals information in Latin acclaim for God. Even though Trithemius was reluctant to publish his work at first, it was finally done so after he died in 1606. Then, the growing steganography in many methods for embedding messages in other media has been developed over time. Microdots, music ciphers, invisible ink, and Morse code on yarn worn by a courier are a few examples. Women spies knitted uneven stitches or purposefully left holes in the cloth to transmit signals throughout both World Wars. The use of personal computers for traditional steganography applications started around 1985.

The information security (InfoSec) [5] field is to protect analog and digital information, not unclassified data. The InfoSec includes steganography, cryptography, mobile computing, social media, steganalysis, organization's infrastructure, networks, etc. Here, we are going to understand some of the recent steganography approaches. According to Touhid Bhuiyan et al. [3], image steganography of the spatial domain is based on the LSB (Least Significant Bit) of the RGB (Red, Green, and Blue) component using XOR (Exclusively-OR) between the 7th and 8th bits. Their approach achieved a higher Peak Signal-to-Noise Ratio (PSNR) and lower MSE (Mean Squared Error), indicating a better subtlety. However, it was limited to payload capacity compared to other methods mentioned and also vulnerable to advanced statistical attacks.

In paper [6], they have overcome the high capacity using stochastic computing in image steganography. They proposed a technique that cover-image pixels and converts them to stochastic streams that can enable the embedding of multiple secret images, and achieve payloads up to 40 bpp. The proposed techniques are based on the algorithm of LSFR (Linear Feedback Shift Register), which utilizes stochastic computing and a probabilistic approach to hide multiple secret messages. However, the overhead from seed image transmission and lower PSNR range from 30-71dB at high payloads. For the secure JPEG (Joint Photographic Experts Group) steganography [16], proposed LSB+ matching and multi-banding algorithmic approach. This approach modifies the JPEG image's DCT (Discrete Cosine Transform) block coefficients. Those DCT coefficients were further divided into four frequency bands and applied the selection strategy coefficient to make the secret message hidden, which leads to less detectability. At embedding, efficiency is decreased, which leads to a loss in statistical preservation.

In paper [19], discusses the wavelet transform domain in steganography for hiding the audio signal of any format into an image. Initially, audio is encrypted using XOR operation and then embedded with approximation coefficients in the DWT

(Discrete Wavelet Transform) of chrominance channels. Finally, at the metric evaluation, the PSNR was 41.6 dB, and the SNR (Signal-to-Noise Ratio) was 38.3 dB. Its limitations include complex implementation with multiple transforms and fixed payload with a size of 512 * 512. Likewise, various classifications for stego images of JPEG-based to the own steganographic methods like OutGuess, Jphide, Steghide, Jsteg, and F5. In [14], a novel algorithm was proposed for multiclass detection using current steganographic methods for JPEG-based images. Their algorithm extracts the 109-dimensional features and trains a SVM (Support Vector Machine) to classify all kinds of stego images and standard JPEG images based on the re-steganography with high precision. The dimensional handling of huge datasets leads to model complications, and the process is too fussy.

For the efficient processing cost and robustness of secure steganography, a new perfect hashing approach was proposed by Imran Sarwar Bajwa et al. [2]. The algorithm was applied to the grayscale image of JPEG and GIF (Graphics Interchange Format) due to their small size. The algorithm process includes a random number for the hash key and an implemented hash function named 'gpref,' which withstands the hash collusion. However, it is slower for large-size images and byte-level secret message embedding. To overcome the byte level to block the pixel. This paper [17] introduced the region selection rule for hiding the secret message. The region considers three factors: HD (Horizontal Difference), VD (Vertical Difference), and RZ (Region Size). This approach can hide the secret message with an approximation of 45% as compared to other methods. However, the identification of the selection block utilizes more computation.

Finally, in the new era of steganography, which utilizes automation models like machine learning, deep learning, etc. Ru Zhang et al. propose [11] a novel CNN (Convolutional Neural Network) architecture named ISGAN (Invisible Steganography Via Generative Adversarial Network) to hide a secret image in the grayscale image into the color cover image from the sender side and revert to the receiver side without any changes. The model performance tests on the LFW, PASCAL-VOC12, and ImageNet datasets. In the process of steganography, the image will lose its pixel values, leading to a loss of image quality information.

We discussed some of the recent research on steganography approaches for undetectable secret messages. The steganography methods are also used for various attacks like image exploits, reverse engineering, etc. by hackers or cyber-attackers [15]. To defend and understand the pattern, steganalysis comes into the picture. In [13], they propose a method for improving the performance of image steganalysis using Siamese networks. The method uses image segmentation and padding to encapsulate both global information and boundaries. These can be employed in four symmetric sub-networks to extract comprehensive steganographic features, image segmented with shared parameters. It is limited to focusing on homogeneous images. To address the challenge of heterogeneous images, the FACSNet (Forensic-Aided Content Selection Network) proposed by Siyuan Huang et al. The author [8] also aims to improve the detection accuracy by up to 7.14% points in tough scenarios by implementing algorithms consisting of a Forensics aided module and content selection module into the FACSnet scheme and utilizing parallel processing of training with steganalyzer to classify the cover images and stego images. However, the performance might depend on the pre-categorization, and complexity may increase due to additional modules.

Looking over the feature selection to improve the efficiency of steganalysis by proposing adaptive feature selection [1]. This addresses the challenges of high time costs and restricted scope in existing feature selection methods for classification metrics. Another view on feature mining and pattern classification is to detect the LSB matching in grey-scale images. This [9] introduces five types of features and compares different learning classifiers. However, this is limited to a lack of fine-tuning and performance variation on different datasets of image types such as grayscale. To enhance the color image, steganalysis [7],[23] proposes a machine-learning approach that makes use of curvelet transformations for feature extraction and SVM for classification. This can achieve high classification accuracy but requires high computational costs for huge datasets.

Utilize the statistical parameter with the fine-tuning model, which results in the robust model for blind approach image steganography. In the paper, [12] uses the statistical features extraction from the GLCM (Gray Level Co-occurrence Matrix) and classifies them using Naive Bayes classifiers. The feature extraction might be expensive in terms of processing units. To overcome that [4,10], work on the adaptive residual extraction and residual co-occurrence probability. Their advantages show higher accuracy and address the limitations of traditional methods. However, they have mentioned in further studies that research to generalize findings over the careful selection parameters.

3. PREPARE YOUR PAPER BEFORE STYLING

The proposed design flow is shown in Fig. 1 and studies a new way of jpeg-based image steganalysis by understanding the selective GLCM features concepts (e.g. energy, entropy contrast, correlation, variance, etc.) gives a novel integrated mathematical improvised evaluation into our proposed blocks of cross-verification using multi-domain correlation and fusion feature extraction process to reducing overfitting, false-positive and false-negative cases. The flow will go with completing the multi-domain correlation cross-verification block task, then the result will be stored in the respective two simulated folders cover_image and stego_image. These simulated folders will be the input for data processing of our whole enhanced classification block of the ensemble stacking. This classification was extracting fusion features of statistical edge-texture, wavelet, and pixel relationships in inputted images and tuning the model. The ensemble classification block utilizes the base classifiers of RF (RandomForest), XGB (eXtreme Gradient Boosting), and LGBM (Light Gradient Boosting Machine) further for optimal stacking ensemble process. For a more detailed block analysis of preprocessing and classification refer to section III (A) and section III (B).

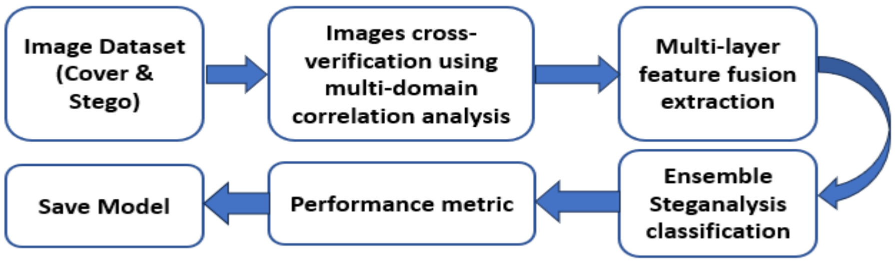
Image Dataset (Cover & Stego)
Images cross-verification using multi-domain correlation analysis
Multi-layer feature fusion extraction
Ensemble Steganalysis classification
Performance metric
Save Model

Fig. 1. Process of proposed block-level steganalysis flow design.

A. Cross- Verification of JPEG Image Preprocessing Using Multi-domain Correlation

Data cross-verification is crucial to our data modeling through multi-domain correlation as a composite anomaly reporting block. The input includes a mix of cover and stego images with different formats like jpg/jpeg, png, etc. Therefore, here data refers to the jpeg/jpg images only and ignores other types of formats using the image validation unit of .jpg or .jpeg format as input of the function. In the initial block, where we concentrated on evaluating the hybrid steganalysis cross-verification of jpg/jpeg formats only.

Before understanding the approach of multi-domain correlation steganalysis part. We need to focus on the input image and understand which channel was used to evaluate. The analysis of images is transformed to grayscale and YUV (Luma, blue-luminance, and red-luminance) channel. From the YUV channel Y channel is also called the luminance channel which represents the intensity of a pixel in an image. This can be used in the multi-domain correlation to analyze the image and finally save it to the simulated storage folders. From here onwards, we were deep-diving into the multi-domain correlation computation. It consists of LSB Variance, Histogram, Metadata inspection, and a hybrid model of DCT+DWT analysis to flag it as a stego anomaly. This can result in the composite anomaly reporting block, which consists of all detected anomalies collected from multi-domain correlation into a log file such as a .csv sheet. Every analysis uses dynamic thresholds based on image properties or observation value of images and represented over individual mathematical representation.

The mathematical formulation represented with the different equation representational numbering and their results in combining with the multiple analysis paradigms to match a unique fingerprint pattern for steganographic flag content. Firstly, in the enhanced LSB analysis stage, instead of just variance, we computed the Shannon entropy of the Y-channel distribution and then combined it with the variance of pixel counts. Therefore, Eq. (1) represents the weighted sum of variance and entropy, compared against an adaptive threshold that uses harmonic mean evaluation of image area in pixels.

$$\Lambda = (2\sigma^2 \cdot H)/(\sigma^2 + H) (1)$$

Where Λ is the LSB anomaly score, σ^2 represents the variance of pixel value counts, H is the Shannon entropy of the Y-channel distribution, and the threshold of the LSB anomaly score should be $\Lambda < 25 \cdot (A/10^6)$. Here $'A'$ it refers to the area of the image in pixels to calculate it. The formula is $A = h * w$, here h, and w are representative image dimensions.

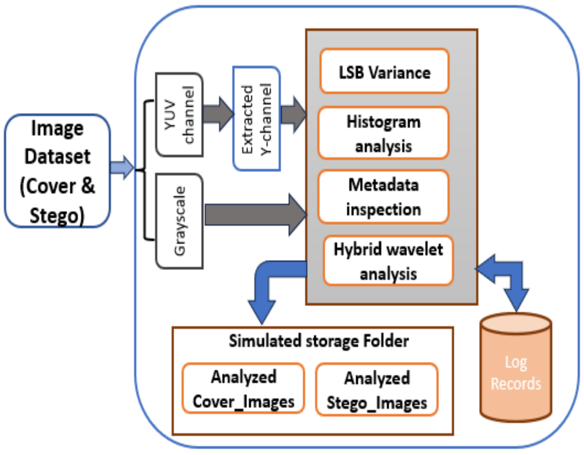

Fig. 2. Images cross-verification using multi-domain correlation analysis block.

For histogram analysis, perform the chi-square test with the observed histogram with the expected uniform distribution (Poisson distribution). Further, adding skewness evaluation to the Poisson test of mean count results in better histogram analysis as shown in Eq. (2).

$$\phi = x^2 + |Y^3| \quad (2)$$

The intermediate calculation of chi-square $x^2 = \Sigma[(h_i - \lambda)^2/\lambda]$ here ϕ represents the divergence metric, x^2 is chi-square, λ represents the Poisson test, Y is the skewness coefficient, and the threshold between $\phi \leq 499 \vee |Y| \geq 2.4$.

In the next stage of analysis, the computation of energy concentration utilization by DWT components with DCT analysis represents the multi-layer frequency analysis. Instead of a traditional standard deviation and variance of individual approaches, here we computed the DWT components ratio of energies between the subbands of LH (Low-High), HL (High-Low), HH (High-High) with the sub-band of LL (Low-Low). Then the kurtosis of the DCT coefficients is calculated and combines the DWT energy ratios shown in Eq. (3). This can lead toward the stego pattern in images with the weighting factors.

$$\Psi = \alpha \cdot ER + \beta \cdot \kappa \qquad (3)$$

The intermediate calculation of the energy ratio $ER = (E\_LH + E\_HL + E\_HH)/E\_LL$. Here α , β are the weighting factors, 0.7 and 0.3. Ψ refers to the multi-layer frequency anomaly index of the DWT component over the DWT energy and κ is the excess kurtosis of DCT coefficients. The threshold between $\Psi \leq 0.14 \vee |\kappa| \geq 4$.

Finally, for improving the overall analysis, the metadata inspection is important to check the multiple fields like EXIF data, comments, and look for unusual strings like binary data extracted using the PIL python package and analyze them as shown in Eq. (4).

$$M = \Sigma[H(m_i)] \quad (4)$$

Here M is the metadata entropy score with each unique frequency list to the length of the number of metadata fields as mathematical representation is $H(m_i) = -\Sigma p_j log_2 p_j$ and threshold should be between $M > 3.5 \cdot N_{meta}$ to flag the anomaly, where N_{meta} is the number of metadata fields obtained from the software stack of PIL package.

To understand the dynamic threshold, let's go with a simple example, usually calculating the average variance of the cover image from the dataset and setting the thresholds relative to that. But here we weren't going through any modeling training phase, this created a complex situation. Alternatively, perform the threshold adjustment based on image size. For the larger images that may have a higher variance, the thresholds for the type of image could be varied and scaled accordingly. However, implementing dynamic thresholds required more complex calculations. To reduce the complexity and keep it simple, adjust the thresholds based on the image dimensions. For a sample example of dynamic threshold implementation, variance threshold = 50 * (total_pixels / 100000). Assuming that an image of 100000 pixels has a threshold of 50, scaled accordingly.

This block is used if the input dataset includes stego and cover images with different formats and is stored in a single folder. So, the multi-domain correlation reduces the complexity of the preprocessing computation at modeling by segregating them into respective folders of cover_dir and stego_dir based on the steganography multi-domain analysis anomaly flag pattern as a result represented in Table 2.

B. Multi-Layer Feature Fusion Extraction

This represents the flow of the classification execution pipeline which loads data of cover images and stego images of JPEG from the III(A) simulated storage respective folders. This pipeline flow includes the extraction of fusion features, stratified train and test splits, Further training of the models, evaluating them, and saving the model (e.g. To test single images, check for the program-level or web interface-level). As we know preprocessing, normalization, and standardization of the dataset images has been done at section III(A).

Firstly, we were looking into the fusion feature extraction process, which initially here reads the images from the simulated storage folder of the cover and stego images. Further, the process at multiple scales, such as the original and the half size, such as resizing to 512*512 using multi-scale processing (s) of 1.0 high quality (HQ) and 0.5 low quality (LQ) of scaled images else reject them if the condition is not met. Each scaled image goes through orthogonal DCT analysis, Benford's law with DCT blocks, dual-tree complex wavelet transforms, and noise residual analysis, then returns all these fused features into the result fusion feature list block. The fusion feature representation is shown below:

For each scale $s \epsilon \{1.0,0.5\}$:

$$F_s = \left[\cup_{k=1}^{N_b} F_{DCT}^{k}\right] \oplus F_{BenfordDivergence} \oplus F_{Wavelet} \oplus F_{NoiseResidual}$$

$$where\ N_b = \left(\frac{W_s}{8}\right) \times \left(\frac{H_s}{8}\right)\ blocks\ per\ scale$$

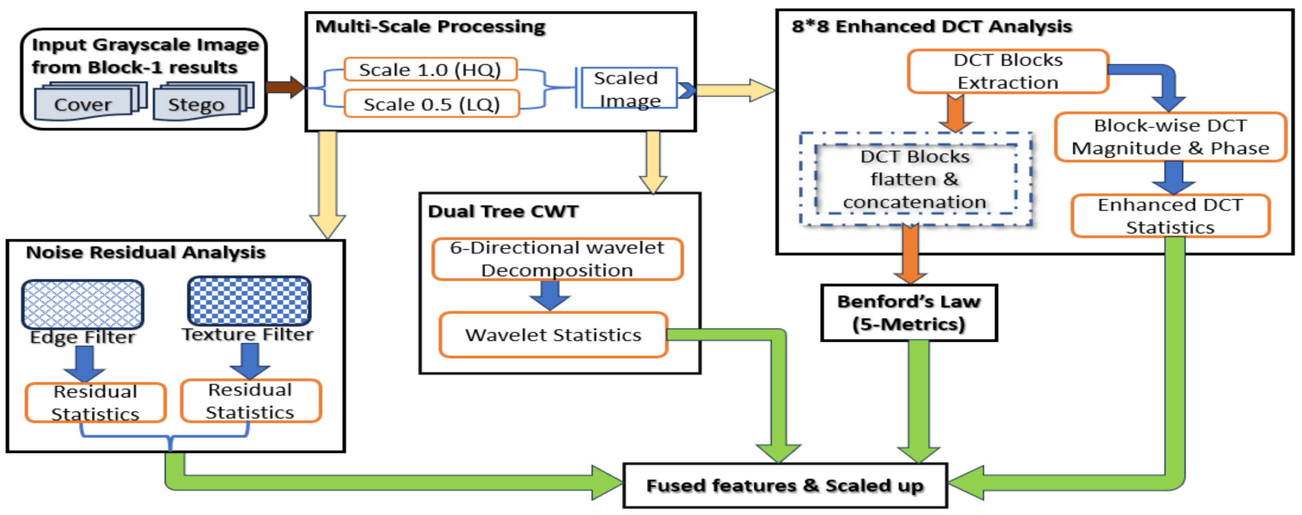

Fig. 3. Multi-scaled image preprocessing and fusion feature extraction.

To understand the scaling up and fusing features, we explained the process of a single image point of view, and it repeats the same for the other images in the respective folders. Firstly, the DCT analysis computes an 8*8 block then DCT represents 2D and is applied twice, once on the transpose, which is equivalent to applying it along both axes of a 2D point of the input image representation. The formula for 2D DCT is well known, but our algorithm uses orthogonal normalization. So, each element in the DCT block was computed as the sum over the image block multiplied by the cosine basis and scaled befittingly which was utilized to extract the magnitude and phase features evaluated through a statical calculation of mean, std, skew, kurtosis, and median phase as some features of DCT analysis. On the other hand, for remaining features, we further developed the novel 2D adaptive DCT with the contrast weighting for the image block using the sigmoid adaptation. Here firstly, the contrast calculation computes the standard deviation (σ) of the image block(I_b) as a contrast. Then, adaptive alpha contrast weighting (α_c) is a sigmoid-like scaling. So, the below mathematical representation α_c would involve the hyperbolic tangent function applied to the contrast normalized by 25 which represents the empirical normalization constant from image dynamics shown in Eq. (5).

$$\alpha_c = 1 + tanh\left(\frac{\sigma(I_b)}{25}\right) \qquad (5)$$

Next, the weighted block is the image block multiplied by α_c. This calculated weighted block will be used to evaluate the directional DCT decomposition of dct_h (horizontal), and dct_hv (both horizontal and vertical) directions. Then find the absolute value of dct_h and the angle of dct_hv for evaluating the phase coherence weighting as shown in the below mathematical Eq. (6-7). Then calculate the weighted magnitude by multiplying the magnitude with phase coherence weighting.

Directional DCT Decomposition:

$$C_{hv}(u,v) = \sum_{y=0}^{7}\left[\sum_{y=0}^{7} \alpha_c I_b(x,y) cos\left(\frac{(2y+1)v\pi}{16}\right)\right] cos\left(\frac{(2y+1)v\pi}{16}\right) \qquad (6)$$

Phase-coherent Magnitude Weighting:

$$M_{weight}(u,v) = |C(u,v| \cdot \left[1 + \frac{1}{2}cos(\phi(u,v))\right] \qquad (7)$$

Here $\phi(u,v) = arg(C(u,v))$

For calculating the remaining features, apply the entropy of phase-entangled magnitude as shown in Eq. (8), and the mean of phase coherence index value represented as Eq. (9), and compute the sum of elements within a weighted magnitude greater than the mean of weighted magnitude. Finally, return all these features into the result fusion feature block.

Phase-Entangled Magnitude Entropy:

$$H_p = -\sum_{u,v} \frac{M_{weight}(u,v)}{M_{weight}} log_2 \frac{M_{weight}(u,v)}{\Sigma M_{weight}} \qquad (8)$$

Phase Coherence Index:

$$\Gamma_\phi = \frac{1}{64}\sum_{u,v}\left[1 + \frac{1}{2}cos(\phi(u,v))\right] \qquad (9)$$

Here phase-magnitude coupling, the $cos(\phi)$ term emphasizes coefficients with aligned phase angles and the modulation factor ε [0.5,1.5]preserves the original magnitude relationships.

After DCT analysis, we computed the Benford divergence with statistical validation Eq. (11), which uses Benford's law. This law says that "The first digit distribution follows $log_{10}(1+1/d)$, here d is the range". This evaluation took the DCT block coefficients to extract the first digits by ignoring the small coefficients (like higher power algebraic coefficient consideration) and checking the input Coeff 'c' for the digit validity check of Benford's divergence evaluation follows as Eq. (10).

First Digit Extraction (FDE):

$$FDE(x) = \left[10^{\{log_{10}x\}}\right] \quad (10)$$

Here, $\{x\} = x - |x|$

$$ValidCoeff(c) = \begin{cases} FDE(|c|) & if\ |c| > 1 \\ excluded & otherwise \end{cases} \quad (11)$$

Then, the observed distribution is computed and compared with the expected distribution using improved KL divergence (Kullback-Leibler) with smoothed probability divergence(D_{SKL}), JS divergence (Jensen-Shannon) with balanced divergence(D_{BJS}), and chi-square evaluated for dimensionally adjusted statistics(χ^2) as represented in equations below Eq. (12-14).

$$D_{SKL} = \sum_{d=1}^{9} \frac{O_d+\epsilon}{T}\, ln\left(\frac{O_d+\epsilon}{T \cdot E_d}\right) \quad (12)$$

Here $\frac{O_d+\epsilon}{T}$ is the smoothed probability with Benford's theoretical probability $E_d = log_{10}\left(1+\frac{1}{d}\right)$. Where O_d is the observed count of leading digit d, $\varepsilon = 10^{-6}$ of smoothing constant for zero-count protection, and total adjusted observations T.

$$D_{BJS} = \frac{1}{2}\left[\sum_{d=1}^{9} \frac{O_d+\varepsilon}{T}\, ln\left(\frac{O_d+\epsilon}{T \cdot E_d}\right) + \sum_{d=1}^{9} E_d ln\left(\frac{E_d}{M_d}\right)\right] \quad (13)$$

The key feature of D_{JS} measuring symmetric treatment of observed and theoretical distributions. The intermediate term $M_d = \frac{1}{2}\left(\frac{O_d+\varepsilon}{T} + E_d\right)$ refers to the midpoint distribution between smoothed observations and Benford's law.

$$\chi^2 = \sum_{d=1}^{9} \frac{(O_d+\epsilon-\mu_d)^2}{\mu_d} \quad (14)$$

Note that unlike traditional χ^2 tests, our formulation uses smoothed observations $O_d + \varepsilon$ and scaled Benford probabilities E_d by the total observations T. This can maintain dimensional consistency between the terms. Here "μ_d" refers to the expected count calculation $\mu_d = E_d \cdot T$.

Further, to calculate dual-logarithmic weighting w(d) which combined upon the information-theoretic $log_2(d+1)$ and Benford-based significance of digit weights represented as Eq. (15).

$$w(d) = log_2(d+1) \cdot log_{10}(1+\frac{1}{d}) \quad (15)$$

Then computes an adaptive significance-weighted null model$(P_{mod}(d))$ where expected digits are scaled by these weights and Benford's law is represented as Eq. (16)

$$P_{mod}(d) = \frac{w(d)}{\Sigma_{k=1}^{9} log_2(k+1) \cdot log_{10}(1+\frac{1}{k})} \quad (16)$$

The original Benford's law is multiplied by its digit weights and normalized for adding two new divergence metrics of weighted cubic (D_{cubic}) and harmonic $(D_{harmonic})$ divergences. The cubic term in the numerator and the denominator with sqrt(n), and the harmonic divergence is the sum of absolute differences over the sum of observed and expected plus one as both represented below as Eq. (17-18).

$$D_{cubic} = \sum_{d=1}^{9} \frac{(O_d - nP_{mod}(d))^3}{nP_{mod}(d)+\sqrt{n}} \quad (17)$$

$$D_{harmonic} = \sum_{d=1}^{9} \frac{|O_d - nP_{mod}(d)|}{O_d + nP_{mod}(d)+\varepsilon} \quad (18)$$

Finally, the statistical measures between observed and predicted probabilities (KL, JS, and chi-squared), cubic, and harmonic divergences were all scaled up to represent our Benford divergence feature result stored into the result fusion feature block.

For DT-CWT (Dual Tree-Complex Wavelet Tree) analysis, A 2D transformer on a scale of images with four levels (nlevels=4). Then it retrieves features from the first four layers' highpass coefficients. The characteristics include the mean absolute value and standard deviation of the angles for each highpass sub-band. Just recall that classic wavelet transformations had problems with shift variance and lack of directionality. Our improved DT-CWT overcomes this by employing two distinct wavelet trees that produce complex coefficients. DT-CWT uses two filter banks for real and imaginary parts. For each level, the transform decomposes the image into multiple directional sub-bands. This provides improved directional sensitivity and shift-invariance.

For an image $Img(x,y)$, the 2D DCWT for each directional subband H_l^d decomposition at level $'l'$, direction $'d'$ produces. The complex coefficients are represented as $C_l^d = R_l^d + jI_l^d$, here R_l^d real tree coefficients and I_l^d imaginary tree coefficients for the directions $d \epsilon \{\pm 15°, \pm 45°, \pm 75°\}$.

Directional Energy Ratio (DER):

$$DER_l^d = \frac{\Sigma|H_l^{(d)}|^2}{\Sigma|H_{l-1}^d|^2+\varepsilon} \cdot tanh(\frac{std(\theta_l^d)}{\pi}) \quad (19)$$

Here, $\frac{\Sigma|H_l^{(d)}|^2}{\Sigma|H_{l-1}^d|^2+\varepsilon}$ is energy scaling & $tanh(\frac{std(\theta_l^d)}{\pi})$ is phase modulation.

Magnitude-Phase Covariance (MPC):

$$MPC_l^d = \frac{1}{N}\Sigma_{(x,y)}(|H_l^d(x,y)| \cdot cos\theta_l^d(x,y)) - (\frac{1}{N}\Sigma|H_l^d|)(\frac{1}{N}\Sigma cos\theta_l^d) \quad (20)$$

Harmonic Phase Coherence (HPC):

$$HPC_l^d = |\frac{1}{N}\Sigma_{(x,y)}(|H_l^d(x,y)| \cdot e^{j2\theta_l^d(x,y)}| \quad (21)$$

Wavelet characteristics in the code are computed for each highpass sub-band. The highpass coefficients from DT-CWT contain magnitude and phase. The characteristics capture the average intensity (mean of magnitudes) and phase variability (standard deviation of angles) for each directional sub-band. The overall feature vector composition is represented in the mathematical formulation as follows below Eq. (22).

For each decomposition level (0-3) and direction (6 directions in DT-CWT):

$$\text{WaveletFeatures} = \cup_{l=0}^{3} \cup_{d=1}^{6} [MPC_l^d, HPC_l^d, DER_l^d] \quad (22)$$

Before understanding the final feature of noise residual analysis. We need to go through the two custom filters of EDGE_FILTER and TEXTURE_FILTER that were designed by us. Let's look into their structure and computation, these are 3*3 kernels sized. Firstly, the EDGE_FILTER has a center of 4 and is surrounded by -2s and 1s. This seems to be like a Laplacian edge detector but with specific weights to highlight the areas of high variance, which might emphasize the edges. The TEXTURE_FILTER has a center of -4, with 2s and -1s around. This could represent capturing the texture variations by returning to a pattern of high frequency.

For EDGE_FLITER 3*3 alias K_{edge}

$$K_{edge} = [[1,-2,1],[-2,4,-2],[1,-2,1]]$$

For TEXTURE_FILTER 3*3 alias $K_{texture}$

$$K_{texture} = [[-1,2,-1],[2,-4,2],[-1,2,-1]]$$

In mathematical terms, applying a filter is equivalent to convolution. Each kernel is convolved with the representing the image and yielding a residual. Applying these filters to an image will highlight edges and textures in the noise residual. Steganographic methods frequently change the noise patterns, hence these filters aid in detecting such abnormalities. They K_{edge} would increase edge-related noise, whereas they $K_{texture}$ would emphasize texture-related noise.

This K_{edge} is different because we consider each point (i,j) in the kernel, we think of the coefficients as a function of their distance from the center. The center value is 4, the neighboring pixels (up, down, left, and right) are -2, and the diagonals are 1. This might be a discrete approximation of the sum of the second derivatives in the x and y axes, along with some diagonal components. Alternatively, it is a scaled-up Laplacian. In comparison to the standard Laplacian kernel $[[0,1,0],[1,-4,1],[0,1,0]]$. Let's know about our K_{edge}, sum of the coefficients: 1-2+1 = 0 for the first row, similarly for the others. The total sum is 0, which is correct for a high-pass filter. The difference is that the K_{edge} are higher absolute values in the corners. So, the K_{edge} kernel is the outer product of the 1D second derivative kernel [1, -2, 1] and itself. K_{edge} is calculated mathematically as v * v^T, where v ranges from 1 to -2. This would improve edges in both directions and their intersections, including diagonals. Thus, the K_{edge} is a 2D second derivative operator that responds strongly to edges in all directions, with the magnitude determined by curvature. The equation is represented as follows below as Eq. (23).

$$K_{edge}(i,j) = v(i) \cdot v(j) \quad (23)$$

They $K_{texture}$ designed to respond to specific texture patterns, such as those with periodic variations. The kernel's structure was a combination of high-pass and band-pass filters. For example, the negative center with positive surroundings in cross directions and negative on diagonals. This enhances areas where there's a transition from the center to the adjacent pixels in horizontal/vertical directions but opposite in diagonals. This $K_{texture}$ is derived from the K_{edge} by applying horizontal and vertical edges over diagonal ones by combining different second derivatives, which capture texture patterns that have more horizontal/vertical components. The mathematical formulation is represented as Eq. (23).

$$K_{texture} = 2 * (K_{horizontal} + K_{vertical}) - (K_{diagonal1} + K_{diagonal2}) \qquad (24)$$

Here $K_{horizontal}$, $K_{vertical}$, $K_{diagonal}$ are the horizontal, vertical, and diagonal second derivatives

To represent the convolved filters with the image as shown below:

$$R_{edge} = I * K_{edge} \qquad (25)$$

$$R_{texture} = I * K_{texture} \qquad (26)$$

The R_{edge} And $R_{texture}$, here R represents as residual and these parameters capture several elements of the noise residual, including overall response, variability, and typical magnitude, which are calculated through respective statistical measures such as mean, standard deviation, and median. Finally, all four analyses will be scaled and concatenated into the result fusion feature block.

C. Steganalysis Classification Modelling

Here we started with the initialization of model components and the hyperparameter grid using the __init__(self) method within the class of the proposed Advanced_Steganalysis_Model. The key components of the __init__ method were base classifier, parameter grids, and special initializations. Let’s look into each component, the base classifier consists of RandomForest, XGB, and LGBM. The parameter grids for each classifier are different. The grid’s hyperparameters will be noted down while testing the models. Special initializations include the best_models to store optimized base models and their optimized parameter grid values, the stack_model enabled the final ensemble model, and the StandardScaler ensured gradient stability and normalized the features extracted from section III(B).

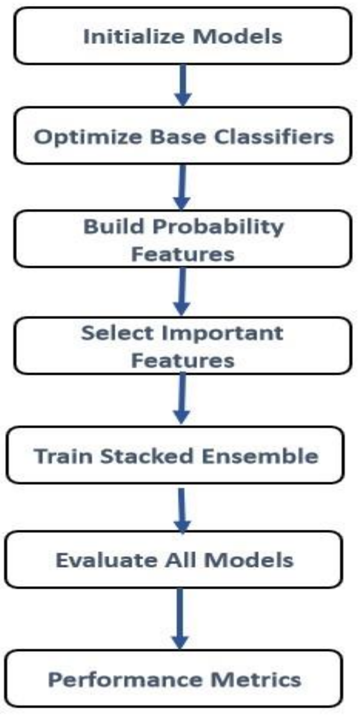

Fig. 4. The flow of our classification model.

The next method was optimize_models (self, X_train, y_train) for optimizing the base models and building the stack_model for the ensemble. The flow of this method starts with cross-validation using StratifiedFold with 7 folds for computational efficiency and maintains class distribution in splits. Then, the hybrid optimization was designed, a combination of

HalvingGridSearch with CalibratedClassifierCV. It uses the HalvingGridSearch, which focuses on precision to reduce false positives and further hyperparameter tuning for each classifier, which is optimized with their respective parameter grid values for resource-aware elimination (reduces search space early and prevents premature stopping). Then the method does the best model calibration using the CalibrationClassifierCV with the sigmoid method and 3-fold calibration to fit the X_train and y_train for better probability estimation and performance maintenance. Next, the stacked feature engineering will be evaluated for the stack_train feature based on the best model's values. This can create the meta-features from the base model probabilities and uses positive loss probabilities only. In the next step, the feature selection is evaluated by RandomForest-based importance calculation using the median threshold, which removes 50% of the least important features and reduces overfitting. The final step in the optimize_models method is ensemble training, which utilizes the stack_model with tuned parameter grid values. This ensemble training uses GradientBoostingClassifier for early stopping with a 20% validation holdout with a tolerance of 0.0001 for loss improvement and a subsample of 89% of data for diversity.

$$P_{stack} = [p_{RF}, p_{XGB,}p_{LGBM}]$$

Here, P_{stack} is probabilities of stack model and $p_{RF}, p_{XGB,}p_{LGBM}$ is the base classifiers probabilities.

For performance analysis, the evaluate_performance (self, X_test,y_test) method is designed, which consists of base model metrics, stacked model processing, and ensemble prediction. In model metrics, precision focuses on minimizing false alarms, recall will ensure the detection of true positives, F1-score, and the AUC score is mapped for the overall ranking capability represented in the V(D) section. The stacked model processing applies the identical transformation pipeline of feature selection and standard scaling in the optimization_models method detailed and explained in the experimental section of V(C). For ensemble evaluation, use calibrated probabilities from the scaled fusion feature of section III(B) and meta-feature from section V(C).

4. PROPOSED TWO-STAGE BLOCK ALGORITHM

In this section, we present the proposed algorithms in two-stage blocks as cross-verification of JPEG images using a multi-domain correlation and multi-layer feature fusion extraction with a novel Ensembling model. The proposed algorithm 1 and 2 follows as below with input, output, and steps.

Algorithm 1: Cross-verification of JPEG images using multi-domain correlation

Input: Given the input of a mixed set of clean and stego images from Kaggle and generated images by Table 1 stego generation protocol.

Output: To obtain the stego_dir and cover_dir images, and the log_files for Algorithm-2 as input of data loading.

Steps:

```
# System initialization
function Cross_Verification_Analysis():
        # input of all images dataset
                img_dir=" path/image_dataset/all"
# Initialize output directory path
if not created then
output_dir=" path/image_outputs"
end if
else
        none/continue
# output directory structure → stego_images & cover_images
# Validation input images with extensions of (.jpg/.jpeg) only into the image_paths
# Initialize the cross-verification analysis log spreadsheet.
        log_file=(output_dir,"cross-verification_analysis_log.xlsx")
# After all required initialization is done, then call the cross-verification image processing
        function image_processing_pipeline (image_paths,output_dir,log_file)  # Function 1.1
```

Function 1.1: image_processing_pipeline(image_paths,output_dir,log_file)

Input: image_paths consists of the dataset path of jpeg/jpg images only. Output_dir is used to store segregated images for the steganalysis pipeline. Log_file with empty tables is used to store the type of image analyzed.

Output: Output_dir is used the segregate input dataset images into the stego_dir and cover_dir images. For evidence preservation, the log_file will be used for model improvements also.

Steps:

```
# Result store path of simulated folders initialization → stego_dir & cover_dir in (output_dir)
# jpg/jpeg file validation unit
# Verify file readability
loop img_path in image_paths: # Check image corruption
        Try:
                img=cv2read(img_path,read_color)
                if img is none then
reject the unprocessed files of img
                end if
        # Preprocessing stage
                Extract y_channel from img_YUV of img
Create a gray image from BGR_GRAY of img
Then calculate the image dimension and compute the area of a pixel of (img)
h,w = img.shape[:2]
area=h*w
        # [Spatial domain probe]
                # LSB variance analysis
                Calculate pixel_distrubtion
                unique_counts = np.unique(y_channel) then find the variance of it.
                pix_distr = unique_counts/(area+1e-10)
                Compute the variance entropy harmonic mean of pix_distr
if adaptive threshold as equation (1) true then
findings.append( LSB_Anomoly )
        end if
# Histogram statistical analysis
Calculate the histogram of the image using cv2
hist=cv2.calchist(img).flatten() then find the hist_diff
Build the intensity distribution of poisson_test, expected of poisson_test
For chi-square to Poisson distribution and skewness analysis, use the equation(2)
if chi-square or abs(hist_skewness) threshold as equation (2) true then
findings.append( Histogram_Anomoly )
        end if
# [Frequency domain scan]
# Wavelet component analysis using DWT subband energies
Decompose (gray_img) using bior 1.3 wavelet
Calculate LL/LH/HL/HH subband
For energy ratio calculation, refer to the equation (3) of (ER) formula
# DCT coefficient analysis
```

```
Computed the wavelet components & mean of deviation refers to the equation (3)
if ER or dct(co_effi) compare against the threshold as equation (3) true then
        findings.append( Frequency_domain_Anomoly )
end if
# [Metadata Inspection]
Extract EXIF/XMP/IMPC data using Image.information
Then analyze binary patterns using UTF-8 to decode the unique values
if meta_entropy compare against mentioned equation (4) payload then
findings.append( Metadata_Anomoly )
        end if
except Raise an Exception for value error type.
# Decision logic sub-block for Anomaly aggregation
analysis_result=''no_anonaly" & flag =''negative'' #initally
Collect all the appended findings and join with the OR decision
if findings then
        analysis_result=" | ".join(findings)
        flag=''positive''
end if
# File categorization
if the flag is positive then
pass (img) into Stego_dir
end if
else
pass the (img) into Clean_dir.
end else
# Evidence previsioning
 #Save the process of image copy with the log
end loop
return the updated (output_dir, log_file) and release the system resources
```

Algorithm 2: Multi-layer feature fusion extraction with novel ensembling model

Input: From the algorithm-1, we get input data for this data loading. Then it evaluates the fusion-based feature extraction and train_test_split the model by using stratified cross-validation. At the final evaluation, the individual models of RF, XGB, LGB, and stack_ensemble_model will undergo the performance metrics after the model training.

Output: The model will be saved at the end of the process using the joblib package for steganalysis classification of the jpeg/jpg input to it.

Steps:

```
function Execute_StegoAnalysis_Pipeline():
#Data_Loading
cover_img_dir="path/simulated_folder/cover_jpeg_images"
stego_img_dir="path/simulated_folder/stego_jpeg_images"
#Fusion_feature_extraction
Initialize the list variable type {X,y} for features and label of images
```

```
loop path, label in {cover and stego directories → 0,1}
loop img_file in {respective_path directory}
    features= Extract_fusion_scaled_features(os.path.join(path,img_file)) # Function 2.1
    if feature is !(None) then
        X.append(features)
        y.append(label)
    end if
end loop
end loop
#list converted to Numpy array (X,y) because it is used for homogeneous data storing
X=np.array(X)
y=np.array(y)
#Train_Test_split
X_train,X_test,y_train,y_test=train_test_split(test=0.3, stratify=y,random_state=42)
#Model training
analyzer=Advanced_steganalysis_Model() # Class 2.1
analyzer.optimize_models(X_train,y_train)
#Evaluation
performance_metrics=analyzer.evalute_performance with the parameter of (X_test,y_test)
finally visualize the metrics with the parameter of (y_test,perfomance_metrics,analyzer)
#For Save model
joblib.dump(analyzer.stack_model, 'Fusion_advance_steganalysis_model.pkl')
#Main function call
if __name__='__main__' then
Execute_StegoAnalysis_Pipeline()
end if
```

Function 2.1: Extract_fusion_scaled_features(img_path)

Input: Image_paths consists of the dataset path of jpeg/jpg images only. This function is used to extract the feature using multi-domain analysis of the statical fusion feature

Output: Finally return the appended feature from four analyses and homogenous those to fuse into one array block.

Steps:

```
Read the image from img_path then apply padding/resize greater or smaller than 512*512
Processed the img for multi-scale processing of scale [1.0,0.5] to obtain scaled_img
#The below 4-analysis in-detailed calculation which represents in 4.2
#Multi-scale DCT analysis
loop over 8 *8 block of scaled_img
dct_blocks ← extract the dct block from img block and store
append the dct_blocks
Extract the feature from magnitude, phase, and novel 2D adaptive DCT with contrasting and weighting
Extend the dct statical feature to fused_feature_list
end loop
#Benford's divergence analysis of dct_block
```

```
Extract the coefficient from the np.concatenate( dct_block.flatten() loop dct_block in dct_blocks)
#Benford's law verification
if coefficients(<100 valid digits) then
        Skips this analysis
end if
else
compute the smoothed KL, balanced JS, dimensional chi-square, and novel divergence metric using the adaptive significance-weighting null model.
end else
Extend Benford's statical feature to fused_feature_list
#Dual tree complex wavelet transform
Extract coefficient for wavelet features by dctwt.transfrom2d of scaled_img with the nlevels=4
Apply wavelet statical on the coefficient highpasses[0-3] of absolute and angle.
Extend the dt-cwt statical feature to fused_feature_list
#Noise Residual analysis
loop kernel in [Edge_filter, Texture_filter]
        residual=cv2.filter2D(scaled_img,kernel)
        Extend noise statical feature to fused_feature_list
end loop
return fused_feature_list
```

Class 2.1: Advanced_steganalysis_Model()

Input: passing the features and labels extracted from function-1 of algorithm-2 as X,y. Further, they split into train-test with the train_size of 0.7 and test_size of 0.3 with the stratified

Output: evaluation of the model with visual metric will be projected in the plot manner which includes model comparisons with precision, recall, and f1-score. Further AUC curve is based on the threshold values for confusion matrix analysis and utilizes important features while tuning the model**.**

Steps:

```
function __init__(self):
#Initialize base classifiers
self.classifiers={ RandomForestClassifier, XGBClassifier, LGBMClassifier}
self.param_grids # Tuned for hyperparameter for each classifier such as n_estimators, learning_rate, etc.
Initializing the self.best_models={}
self.stack_model=None # stacking initialization for Ensemble
self.scaler = initialization with StandardScaler
function optimize_models(self, X_train,y_tarin):
skf= StratifiedKFold(splits=7,Shuffle=true)
optimized_params={'RF',' XGB', LGBM'} # optimized Tuned for hyperparameter for each classifier such as n_estimators, learning_rate, etc.
loop name in self.classifies:
        # faster search HalvingGridSearchCV used
search= HalvingGridSearchCV( self.classifiers[name], optimized_params/param_grids [name],cv=skf,scoring='precision',njobs=-1,aggressive_elimination=True)
Search.fit(X_train,y_train)
```

```
self.best_models[name]=calibratedClassifierCV(search.best_estimator_,cv=3, method='sigmoid').fit(X_train,y_train)
end loop
# Gradient Boosted Stacking with Early Stopping
stack_train=column_stack([ model.predict_proba(X_train)[:,1] loop model in self.best_models.values()])
# Feature Selection for stacking
self.selector=SelectFromModel(RandomForestClassifier(n_estimators=100, random_state=42
),threshold='median').fit(stack_train,y_train)
stack_train_scaled= selector.transform(stack_train)
# Early-stopped Gradient boosting parameter
self.stack_model=GradientBoostingClassifier( parameters)
stack_model.fit(stack_train_scaled,y_train)
Then perform to Prune the unused estimators for stack_model.set_params
#Function to evaluate the evaluate_performance(self,X_test,y_test) for precision,recall,F1-score, and AUC visualize them on plot.
#Plot the feature importance analysis (RandomForest)
```

5. EXPERIMENT & RESULTS

A. Dataset Design

From the Kaggle website, the public dataset of images consists of 500_cover_images and 500_stego_images. The remaining images are generated using the below-mentioned methods in Table 1. For cover images, we have downloaded from our local machine's Omen wallpaper application with the resolution stratification of 512*512. So, these cover images undergo the stego generation protocol of embedding different methods such as LSB replacement, DCT coefficient, wavelet domain, and metadata injection.

TABLE 1. Stego image generation data

Embedding Method	**Tool**	**Payload Density**	**Samples input of cover images**	**Stego images generated**
LSB Replacement	Steghide v1.3	0.1-1.0 bpp	200	200
DCT coefficient	F5 v2.2	QF=75, 50% capacity	100	100
Wavelet Domain	OutGuess 0.2	3-level DWT embedding	100	100
Metadata Injection	ExifTool 12.6	Random binary payloads	100	100

B. Parameter Space Exploration for Cross-verification Analysis Block

After completing the dataset preparation, the algorithm-1 and its functions are written in the Python script that uses various image processing techniques to cross-verify the dataset of images to simulate them into the respective folders of cover and stego as the aim. Here, this experiment checks for LSB anomalies, histogram abnormalities, frequency domain anomalies, and metadata entropy for jpeg/jpg images. The main function (algorithm 1) images from the directory, analyzes them, and categorizes them further, logging the results. In this, we proposed the adaptive threshold for LSB anomalies and DWT components on the DCT for the multi-layer frequency analysis. As per our testing, we consider the below-mentioned threshold sensitivity analysis and wavelet configuration testing, these combinations are called parameter space exploration as shown in Table 2. This experiment tests these thresholds to see how changing these values affects cross-verification with prior merged into one folder of cover and stego images before and after testing for correctly cross-verified . This will be used during the pre-classification of the model to improve the overall steganalysis classification model with the best-fit model and better F1 score.

TABLE 2. Parameter Space Exploration

Type of Analysis	**Threshold values**	**Before cross-verfication images are available in a separate folder count.**		**After merging the available images into one folder, and cross-verify them correctly, then segregate them into the respective folders.**		**Consideration for the cross-verified images for the ensemble classification model.**
		Cover	***Stego***	***Cover***	***Stego***	
LSB coefficient	15	100	100	70	60	✗
	25			91	93	✓
	35			85	90	✗
Chi-square limit	300	100	100	73	69	✗
	500			90	94	✓
	700			82	88	✗
DCT kurtosis bound	3	100	100	80	76	✗
	5			89	92	✓
	7			72	69	✗
Meta Pattern	2.5	100	100	85	79	✗
	3.5			92	90	✓
	4.5			82	80	✗
Wavelet types {bior1.3, haar, db4, sym5} (DL= Decomposition Level)	DL = 1	100	100	73	69	✗
	DL = 2			82	86	✗

	DL = 3			93	96	✓
The public dataset of Kaggle undergoes the above analysis	Considered ✓ threshold values	500	500	495	498	✓

C. Classifier Models Training and Ensemble Stacking Analysis

From Table 2 we selected the image curation as protocol to proceed with the input to algorithm 2. The Sources include cover_images and stego_images which already strict image validation adheres to the exclusion of monochrome/synthetic images using entropy thresholding because these can't possess actual steganalysis in a real-time environment. So, real-time images are taken into consideration, and then the processing stage of all images is converted to 8-bit grayscale matrices in 6.2. But here we are multi-scale by applying a pad of non-512*512 images with mirrored content. To maintain the tile grid of 8*8 and clip limit to 2.0 by applying CLAHE for illumination normalization. Then feature extraction of JPEG images which includes DCT analysis per 8*8 block, Benford's law compliance analysis check, Dual tree-CWT analysis, and noise residual analysis represented at 4.2. These fusion features will be split for a train-test of 70-30 stratified which maintains 35% stego in both sets using cryptographic RNG seeding. The model ran on the hardware configuration of NVIDIA RTX 4050 GPU with CUDA 11.2, 16GB of DDR5 RAM with zRAM swap compression, and storage of 1TB NVMe space, for the memory-efficient workflow batch processing is used to extract features in 100-image chunks and utilize the data type of float32 for features of native GPU precision synchronization. To avoid the isolation processing unit, we used the concept to implement the parallelism of 16 CPU threads for feature extraction with the 1 GPU stream per model.

In algorithm 2 analyzer calls the *Advanced_steganalysis_Model()* (class 1) for optimized model training and evaluation. But in this section, we are going to understand the model classifier tuning. Initially, we need to prepare three tree-based-classifiers of RandomForest (RF), XGBoost (XGB), and LightGBM (LGBM). Here we designed different classifiers in such a way that can be utilized in our proposed algorithm. The RF uses class weighting to handle imbalanced datasets with the hyperparameter search design composite of 48 combinations (3*2*4*2 grid). The XGB is optimized for GPU accelerated with the hyperparameter search design composite of 36 combinations with early pruning. The LGBM implements gradient boosting with leaf-wise growth with the hyperparameter search design composite of 24 combinations by using Bayesian optimization. The overall hyperparameter search design was placed with the termination criteria of <1% precision improvement over 5 iterations and GPU memory usage >80%. The custom hypermeter search design for each model varies in depth (10-15), and subsampling rates (0.7-0.9) and balances the overall model complexity (n_estimates:200-500) with training time. For resource-efficient search we used HalvingGridSearch instead of standard grid search because three factors progressively eliminate weak parameter combinations, allocate more resources to promising job candidates, and which an aggressive_elimation parameter for faster pruning.

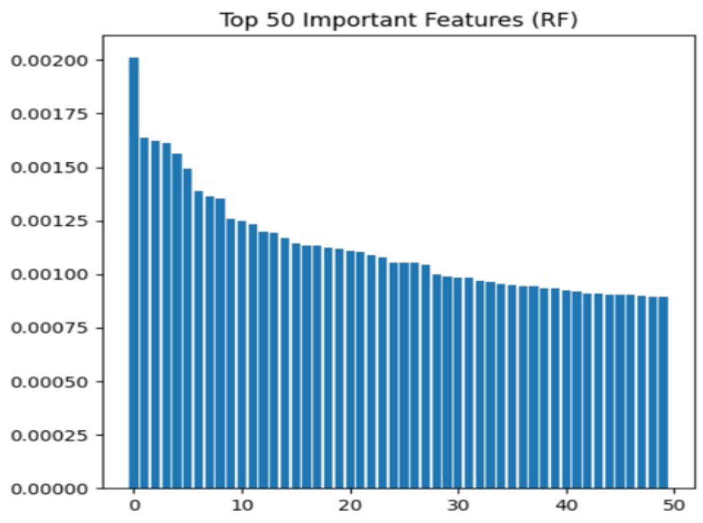

Fig. 5. Stacking model's important feature selection

Here we are constructing the meta-model construction using four components meta-feature creation, feature selection, feature scaling, and GradientBoosted meta-learner. Firstly, the meta-feature creation collects class probabilities (column 1) from all based classifier models and forms a new dataset of N-samples * features of one per classifier. Then feature selection uses the SelectFromModel with RF to retain the features with an importance ≥ median value which leads to reduced dimensionality while preserving the signal as shown in Fig. 5. As the third component feature scaling applies StandardScaler to normalize meta-features and is critical for gradient-based optimization in subsequent steps. The Gradient Boosted Meta-Leaner initialized by GradientBosstingClassifer with a large tree count of 2000 for capacity. The n_iter_no_changes=50 is early stopping to prevent overfitting in the model and shallow trees (max_depth=3) for smooth decision boundaries. This Gradient-boosted meta-learner automatically determines the optimal stopping point and prunes the excess tree post-training for efficiency.

D. Models Evaluation and Comparsions

In algorithm 2 the evaluate_performance is used to calculate the base model's assessment for each classifier (RF, XGB, LGBM), ensemble model validation which generates the class predictions, and calculates the precision (P_m), and recall(R_m) to evaluate F1-score as shown in Fig. 6, and AUC_ROC of thresholds shown in Fig 7 for the performances. In steganalysis, both FP (cover images labeled as stego) and FN (stego images labeled as clean) can have significant consequences. So, the F1- score is used to strike a balance between these two errors. We need to understand further the impacts on balanced and imbalanced datasets given to our model as input. In the view of balanced datasets precision and recall are equally important which means a good model balance both (e.g. F1-score). Whereas imbalance datasets possess high precision and low recall which refers to common when the minority class is rare/lower and implies the false positives are costly. The other case for recall is low precision and high recall which refers to when missing positives are worse than false indications[20]. Here our model evaluates for a balanced dataset but the RF model is built for an imbalanced dataset and the other two models are strictly built for the balanced dataset. This type of ensemble model will help to steganalysis model give best-fit results at the end as shown in Table 3.

$$P_m = \frac{TP}{TP+FP} \qquad (27)$$

$$R_m = \frac{TP}{TP+FN} \qquad (28)$$

Where TP, FP, FN, and TN represent True Positive, False Positive, False Negative, and True Negative are the basic terms of the confusion matrix terms. Here the "m" represents the models.

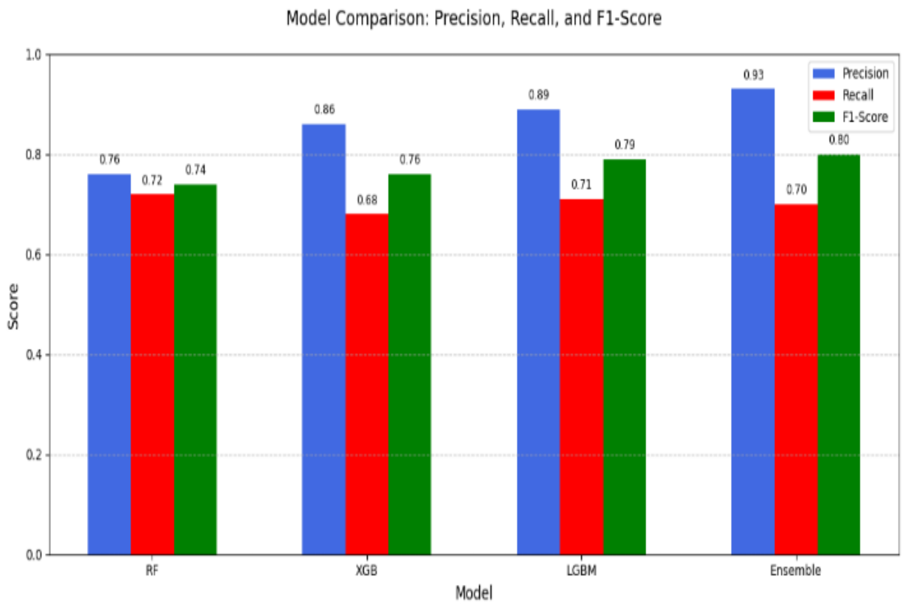

Fig. 5. ***Comparison of models.***

$$F1_{score} = 2 \times \left(\frac{P_m \times R_m}{P_m + R_m}\right) \quad (29)$$

Here, $F1_{score}$ is the harmonic mean of P_m & R_m. If F1 is high, it directly represents a balanced model (values closer to 1 are better than the model).

TABLE 3. *Model-Specific Analysis*

Our Custom Model	Fig. 6 comparison represents an optimized parameter grid value test with some variation in different devices.			Strength	Weakness
	Precisi-on	**Recall**	**F1-score**		
RF	0.76±0.07	0.72±0.07	0.74±0.04	Best recall & good F1	Moderate precision
XGB	0.86±0.05	0.68±0.06	0.76±0.02	Balanced F1 & good for general use	Slightly lower recall than RF
LGBM	0.89±0.02	0.71±0.04	0.79±0.03	High precision but fewer False Positive	Weakest recall and F1
Ensemble/Stack Ensemble	0.93±0.01	0.70±0.03	0.80±0.01	Robust precision and stable predictions due to high F1	Slight requires more resources

As Fig 7 represents the thresholds of axes X & Y are TP and FP rates. These threshold adjustments represented for classification thresholds were lowered to reduce false negatives, directly boosting recall [21],[22]. To understand, we derive a relationship connecting AUC (computed from two threshold rates) to precision, recall, and F1-score using geometric and harmonic principles. Threshold 1 has FPR=0.0 and TPR=0.0 which is a use case to avoid all false positives (FPR=0) but fails to detect any true positives (TPR=0). Whereas threshold 2 of models defines the relaxed threshold allowing some FP/TP trade-off and balancing the steganalysis (TPR) and false alarms indication (FPR).

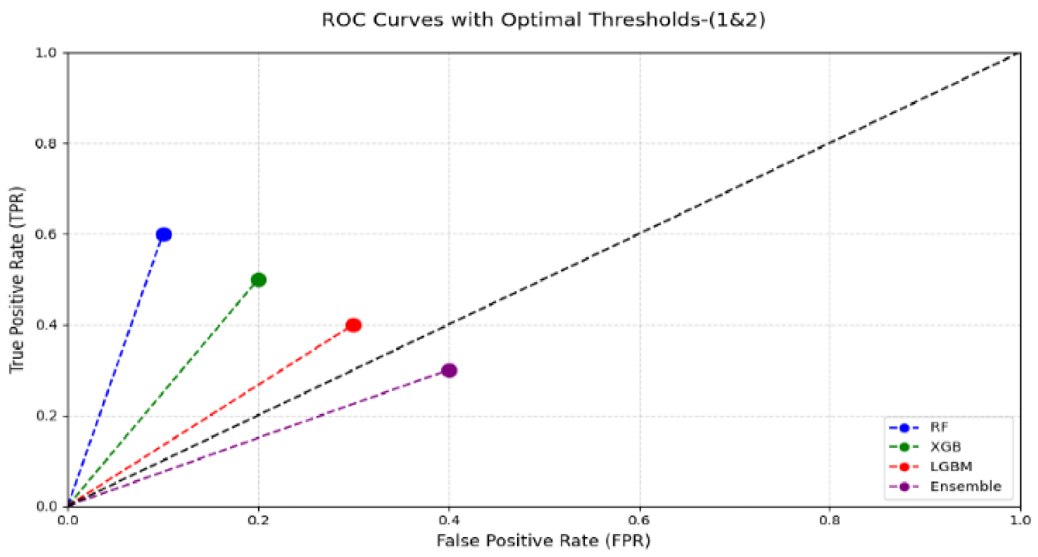

Fig. 6. Optimal thresholds of 1 and 2 for model performance analysis.

The derivation for thresholds 1 and 2 (triangle for 2 points) with the geometric relationship of AUC is the area of the triangle formed by (0,0), (FPR_2,0), and (FPR_2,TPR_2).

$$AUC_m = \frac{1}{2} \times FPR_2 \times TPR_2 \qquad (30)$$

Linking AUC_m to $F1_{score}$, by using $F1_{score}$ the formula, substituting $R = TPR_2$, rewrite Eq. (27) as $P_m = \frac{F1_{score} \times R}{2R - F1_{score}}$, and substitute $TPR_2 = R$ in Eq. (30). Then $AUC_m = \frac{1}{2} \times FPR_2 \times R$ rewrite as $FPR_2 = \frac{2 \times AUC_m}{R}$. Therefore, the final relationship of Eq. (30) after deriving is represented below with Eq. (31).

$$AUC_m = \frac{FPR_2 \times F1 \times R}{2(2R - F1_{score})} \qquad (31)$$

Here the model is evaluated to two thresholds, where we obtained to better F1 score analysis for model evaluation and performance. The interpretation of AUC_m vs $F1_{score}$ indicates that higher AUC_m requires either higher FPR_2 or TPR_2, but this conflicts with precision $\left(P_m \ \alpha \ \frac{1}{FPR_2}\right)$. As the trade-off to optimizing the $F1_{score}$ balances P_m and R_m, while AUC_m penalizes high FPR_2.

6. CONCLUSION AND FUTURE WORK

We conclude that the stego-based image exploits make the data modeling vulnerable and the area of other fields also. To overcome that we proposed an approach of two-block integrated algorithms consisting of cross-verification images using multi-domain correlation and improvised stacked ensemble classification for steganalysis. Here the block works to separate for image preprocessing and utilizes the image properties to calculator the statical, structural, and frequency domain to segregate the images into simulated folders of cover and stego directory. This process reduces computation resources by utilizing the multi-domain and their mathematical evaluation. Then come to the block of stacked ensemble classification utilizing the base classifiers. Here we designed approaches as follows firstly, the initial processing image undergoes through multi-scale processing to capture the artifact at varying resolutions.

The hybrid logarithmic weighting, phase-entangled magnitude, non-linear context adaption, cross-level energy tracking between adjacent scales, the custom filters for noise residual, and many more enhance the fusion features. Then the model evaluation we used the efficient techniques of halvingGridSearch and GPU rendering with robust statical practices including calibrations classifier, stratification specially designed for steganalysis unique challenge of the subtle signal of high dimensional spaces, and probability refinements. The dimensionality control uses median RF feature selection to reduce the overfitting and standard scaling ensuring the gradient boosted for meta-featuring. Here the model is a limited to moderate dataset and the two thresholds we evaluated to better F1 score analysis for model evaluation and performance. We can also take more thresholds based on the above mathematical calculation equation. Our proposed algorithms though have significant changes to reduce computational resources and build for balanced datasets to handle small variations in imbalance datasets only processed as input to have stable prediction in our novel ensemble modeling. This can lead to one of the approaches in the machine learning model for the steganalysis field. To improve the limitations in the future we are going to work with the deep learning model to adapt more fusion features, improve threshold optimization, and more to handle the imbalance dataset of other image formats also.

REFERENCES

[1] Y. Ma, X. Yu, X. Luo, D. Liu, and Y. Zhang, “Adaptive feature selection for image steganalysis based on classification metrics,” *Information Sciences*, vol. 644, p. 118973, Apr. 2023, doi: 10.1016/j.ins.2023.118973.

[2] I. S. Bajwa and R. Riasat, "A new perfect hashing based approach for secure stegnograph," 2011 Sixth International Conference on Digital Information Management, Melbourne, VIC, Australia, 2011, pp. 174-178, doi: 10.1109/ICDIM.2011.6093325.

[3] T. Bhuiyan, A. H. Sarower, R. Karim, and M. Hassan, “An Image Steganography Algorithm using LSB Replacement through XOR Substitution,” *2018 International Conference on Information and Communications Technology (ICOIACT)*, pp. 44–49, Jul. 2019, doi: 10.1109/icoiact46704.2019.8938486.

[4] L. Wang, Y. Xu, L. Zhai, Y. Ren, and B. Du, “A posterior evaluation algorithm of steganalysis accuracy inspired by residual co-occurrence probability,” *Pattern Recognition*, vol. 87, pp. 106–117, Oct. 2018, doi: 10.1016/j.patcog.2018.10.003.

[5] A. Zanke, T. Weber, P. Dornheim, and M. Engel, “Assessing information security culture: A mixed-methods approach to navigating challenges in international corporate IT departments,” *Computers & Security*, vol. 144, p. 103938, Jun. 2024, doi: 10.1016/j.cose.2024.103938.

[6] M. El-Hady, M. H. Abbas, F. A. Khanday, L. A. Said, and A. G. Radwan, “DISH: Digital image steganography using stochastic-computing with high-capacity,” *Multimedia Tools and Applications*, vol. 83, no. 25, pp. 66033–66048, Jan. 2024, doi: 10.1007/s11042-023-17998-9.

[7] Kodati, S., Reddy, K.P., Ravi, G., Sreekanth, N. (2022). IoT-based System for Health Monitoring of Arrhythmia Patients Using Machine Learning Classification Techniques. In: Reddy, V.S., Prasad, V.K., Wang, J., Reddy, K. (eds) Soft Computing and Signal Processing. ICSCSP 2021. Advances in Intelligent Systems and Computing, vol 1413. Springer, Singapore. https://doi.org/10.1007/978-981-16-7088-6_25

[8] S. Huang, M. Zhang, Y. Kong, Y. Ke, and F. Di, “FACSNet: Forensics aided content selection network for heterogeneous image steganalysis,” *Scientific Reports*, vol. 14, no. 1, Nov. 2024, doi: 10.1038/s41598-024-

77552-x.

[9] Q. Liu, A. H. Sung, Z. Chen, and J. Xu, “Feature mining and pattern classification for steganalysis of LSB matching steganography in grayscale images,” *Pattern Recognition*, vol. 41, no. 1, pp. 56–66, Jun. 2007, doi: 10.1016/j.patcog.2007.06.005.

[10] J. Li, X. Wang, Y. Song, and P. Wang, “FPFnet: Image steganalysis model based on adaptive residual extraction and feature pyramid fusion,” *Multimedia Tools and Applications*, vol. 83, no. 16, pp. 48539–48561, Nov. 2023, doi: 10.1007/s11042-023-17592-z.

[11] R. Zhang, S. Dong, and J. Liu, “Invisible steganography via generative adversarial networks,” *Multimedia Tools and Applications*, vol. 78, no. 7, pp. 8559–8575, Dec. 2018, doi: 10.1007/s11042-018-6951-z.

[12] A. Aljarf, H. Zamzami, and A. Gutub, “Is blind image steganalysis practical using feature-based classification?,” *Multimedia Tools and Applications*, vol. 83, no. 2, pp. 4579–4612, May 2023, doi: 10.1007/s11042-023-15682-6.

[13] L. Fan, J. Qiu, Z. Wang, and H. Wang, “Maximizing steganalysis performance using siamese networks for image,” *Multimedia Tools and Applications*, vol. 83, no. 31, pp. 76953–76962, Feb. 2024, doi: 10.1007/s11042-024-18572-7.

[14] Xiaozhong Pan, BoTao Yan and Ke Niu, "Multiclass detect of current steganographic methods for JPEG format based re-stegnography," 2010 2nd International Conference on Advanced Computer Control, Shenyang, 2010, pp. 79-82, doi: 10.1109/ICACC.2010.5486869.

[15] R. Böhme, “Principles of Modern Steganography and Steganalysis,” in *Information security and cryptography*, 2010, pp. 11–77. doi: 10.1007/978-3-642-14313-7_2.

[16] H. -t. Wu and J. Huang, "Secure JPEG steganography by LSB+ matching and multi-band embedding," 2011 18th IEEE International Conference on Image Processing, Brussels, Belgium, 2011, pp. 2737-2740, doi: 10.1109/ICIP.2011.6116235.

[17] V. Saravanan and A. Neeraja, "Security issues in computer networks and stegnography," 2013 7th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, India, 2013, pp. 363-366, doi: 10.1109/ISCO.2013.6481180

[18] J. Reeds, “SOLVED: THE CIPHERS IN BOOK III OF TRITHEMIUS’S STEGANOGRAPHIA,” *Cryptologia*, vol. 22, no. 4, pp. 291–317, Oct. 1998, doi: 10.1080/0161-119891886948.

[19] S. Hemalatha, U. D. Acharya, and A. Renuka, “Wavelet transform based steganography technique to hide audio signals in image,” *Procedia Computer Science*, vol. 47, pp. 272–281, Jan. 2015, doi: 10.1016/j.procs.2015.03.207.

[20] Dineshkumar, R., Siddhanti, P., Kodati, S., Shnain, A. H., & Malathy, V. (2024). Misbehavior detection for position falsification attacks in VANETs using ensemble machine learning. In *IEEE* (pp. 1–5). https://doi.org/10.1109/icdsis61070.2024.10594423.

[21] Kalpana, P., Kodati, S., Sreekanth, N., Ali, H. M., & C, R. A. (2024, August 23). Predictive Analytics for crime prevention in smart cities using Machine Learning. In *IEEE*. https://doi.org/10.1109/iacis61494.2024.10721948.

[22] Jayasingh, B. B., Kumar, B. S., & Mallareddy, A. (2024, October 24). *Predictive Analytics Model for Heart Disease: Leveraging Machine Learning Techniques*. https://doi.org/10.1109/icicec62498.2024.10808569.

[23] Reddy, K. P., Ramakrishna, C., Reddy, V. S., Veeranna, T., Kodati, S., & Benarji, T. (2025). Implementation of predicting diabetes disease using machine learning based unified framework. In *Cognitive science and technology* (pp. 15–24). https://doi.org/10.1007/978-981-97-8533-9_2.