I.J. Image, Graphics and Signal Processing, **2026, 4, 25-47**
Published Online on August 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2026.04.02

A Multi-Branch Transformer Based Cross-Attention Framework for Computer-Generated Image Detection

Venkata Satya Renuka Devi Bhamidipati*
Department of Electronics & Communication Engineering, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India
E-mail: bvsrenuka@gvpcew.ac.in
ORCID iD: https://orcid.org/0009-0002-5843-9115
*Corresponding author

Srinivasa Rao Chanamallu
Department of Electronics & Communication Engineering, Jawaharlal Nehru Technological University Kakinada, University College of Engineering, Narasaraopet, India
E-mail: chsrao.ece@jntukucev.ac.in
ORCID iD: https://orcid.org/0000-0001-6373-8342

Sudheer Gopinathan
Department of Mathematics, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, India
E-mail: g.sudheer@gvpcew.ac.in
ORCID iD: https://orcid.org/0000-0001-9365-4705

Received: December 22, 2025; Revised: January 13, 2026; Accepted: February 18, 2026; Published: August 8, 2026

Abstract: The rapid development in deep learning-based generative software and image rendering tools has led to the generation of massive photorealistic digital content – fake images, fake videos, fake speech, etc. Such fake digital media may result in communication of misinformation, forgery of digital data, and eroding trust in the information source. This poses a significant challenge to the field of techniques in digital forensics, to which our present work attempts to make a contribution, by addressing the problem of differentiating AI-generated images from real photographs, using transfer learning and a multi-branch fusion model. We propose a multi-branch model that integrates two pre-trained Vision Transformer models (DINO (self-distillation with no labels) and Contrastive Language – Image Pretraining (CLIP)) to extract complementary global features, along with a forensic and a hand-crafted feature branch, which extract low-level discriminating cues. These features are complementary to each other and hence contribute to improvingthe robustness and performance of the model. These features from the four branches are adaptively weighted and combined by a cross-attention module, to give a fused and rich embedding. The model is further optimized by using augmentation-invariant loss, center loss and supervised contrastive loss in addition to the cross-entropy loss function. This framework achieves improved accuracy of 95.83% on PRCG dataset, 96.79% on CIFAKE dataset and 99.15% on GenImage dataset as compared to baselines. It also achieved stable cross-generator performance and enhanced robustness against real-world corruptions like Blur, Noise, Compression, and others. The experimental results show a good separability between the classes, and enhanced performance on publicly available datasets.

Index Terms: Computer-Generated Image, Computer Graphics, Image Forensics, Fake Image Detection.

1. Introduction

Digital content has become a very important tool in information exchange in the present world. Digital images and videos have wide applications in the fields of medicine, science, technology and entertainment among others. These digital media may either be obtained by a digital camera also called natural images (NI), or might be computer-generated images (CGI). The CGIs may be obtained by graphic tools, image rendering software or deep-learning algorithms like Generative Adversarial Networks (GAN) [1]. The present-day image rendering software applications

and generative models have developed so much that, differentiating a fake image/video from a real one is almost impossible for the naked eye. Fig. 1 shows samples of fake and real images from different publicly available datasets.

Fig. 1. Samples of fake and real images. The top row shows real images, and the bottom row shows fake images from datasets like GenImage, and PRCG.

According to Zeyu Lu et al, humans struggle to detect fake images and differentiate them from real images, with a misclassification rate of 37% [2]. Although the computer graphic images and videos have wide applications in gaming, entertainment, art and many other industries, they are also put to malicious and illegal use in many areas, thus raising questions on trust and authenticity of digital media. This leads to a growth in demand for image forensics, which analyzes digital images for authenticity, manipulation and other details.

Many algorithms were developed by researchers to authenticate digital images. Existing methods use handcrafted features, and deep learning techniques to address the task [3,4]. Many methods use transfer learning, multi-branch methods and initialization methods to improve the performance and generalizability of the models to unseen generators [5,6]. Few researchers have also developed self-supervised methods to handle insufficiency of labelled data [7]. However, ample scope for improvement in performance and reliability of this authentication process, still exists and is achievable, which is demonstrated by our proposed algorithm. The rest of the paper is organized as follows: section 2 outlines the relevant literature. Section 3 discusses the proposed model in detail. Section 4 analyzes the performance of the model and the comparison with other baselines. Section 5 concludes the paper, and suggests the future scope.

2. Related Work and Motivation

The problem of distinguishing AI-generated images from real photographs is an important area in the field of image forensics. This has attracted a lot of research in this area. This section presents a brief review of the work done in this area. Earlier works in this area include models based on manually derived features and models based on deep learning.

The first category depends on manually designed features from either spatial domain or transform domain. These features may include shape, color, texture etc. For instance, Tian-Tsong Ng et al developed a geometry-based model for differentiating real and AI-generated images based on the physical differences between the two classes, by using fractal and differential geometry at different scales of the image [8]. However, the limitation of the handcrafted feature-based methods is that it is slightly laborious, and the performance may be low on complex and challenging datasets.

On the other hand, the deep learning-based methods use deep neural networks to learn features required for addressing the task. Ruisong Zhang et al used correlation between the color channels and pixel level correlation to solve the problem of demarcating AI-generated and photographic images [9]. The authors used a hybrid module along with CNN models that extracts correlations and uses this information to improve their performance. Ivan Castillo Camacho et al proposed two approaches for the initialization of first layer of a CNN – one data-dependent, and other data-independent – which helped in achieving variance stability at the output of the convolution filter [6]. Both the approaches led to improvement in CNN-based image manipulation detection methods. Chanthini Baskar et al used the transfer learning approach to differentiate CG and Natural images [5]. They modified the input layer, dense layer and the prediction layer of the pre-trained models and fine-tuned them. By comparing the validation accuracy of all these models, they chose the best model (SqueezeNet) for transfer learning. Manjary P Gangan et al approached the problem of differentiating Computer graphics, GAN generated and natural images, by fusing three EfficientNet networks each operating in a different colorspace [10]. Apart from the fully supervised methods like the ones described above, Kai Wang proposed a self-supervised learning method with forensic oriented pretext task, to differentiate authentic images from their manipulated versions [7].

Another problem to be tackled in the area of computer-generated image detection is the task of generalization or classifying fake images generated by unseen generators. In this direction, Weize Quan addressed the task of blind detection of computer-generated images [11]. In order to improve the performance of the network on images from unseen generators, the authors used a network with two branches, each with a different initialization for its first layer, to extract discriminating features. For enhanced training, they introduced negative sample generation in two ways-one by interpolating unpaired real and fake image pairs, and second by modifying the fake image based on the gradient of CNN

model. Patrick Grommelt et al demonstrated in their paper that datasets used for AI-generated image detection tasks, display biases due to imperfections that arise in images due to compression, and size of image. These result in misaligned evaluations [12]. They showed that by applying constraints on these datasets in order to mitigate these biases, there was a substantial improvement in robustness and generalization.

The recent works that address the problem of forgery detection and localization include transformer-based models. For instance, Guillaro et al., developed a forensic framework, TruFor that uses noise-sensitive fingerprint to capture artifacts introduced due to camera model, as well as due to processing like editing, by training only on real pristine photographs [13]. The model detects forgeries as deviations from the expected real image patterns. Although it can locate unknown manipulations in the image, it requires pixel-level supervision, and cannot detect fully generated images. On the other hand, Wang et al., addressed the problem of generalization of fake image detection by developing a universal fake image detection framework, UniFakeD, that can be used for artificially edited images, as well as deep model generated images via multiple ensemble learning method (inner-set and inter-set ensembles) [14]. The model's generalization capability on unknown data is improved with the help of adaptive voting mechanism. Jeong et al., developed a deepfake image detection model, that can generalize over unseen GAN models [15]. Their framework used FrePGAN which generates frequency-level perturbation maps which are added to the images, which help in reducing the effect of domain-specific artifacts. They also used a deepfake classifier which was trained along with the FrePGAN module alternatively to consider both frequency-level and pixel-level features. The problem of generalization over unseen models was also addressed by Tan et al, whose model, LGrad, extracted gradients through a pre-trained CNN based transformation model, which were used as the generalized representation of the image [16]. This was based on the fact that, gradients are more dependent on the pretrained model, and less on the training sources.

However, as the AI generator technology emerges, there exist lot of challenges to be addressed in the field of digital forensics. Every image holds some inherent cues and leaves traces of its source. A photograph, whether acquired by a camera, or generated by graphics, leaves fingerprints, which if tapped can be used to address the problem at hand. These fingerprints may be in the form of global features, or pixel-level features. Transformers, which are based on attention-mechanism, are capable of capturing holistic features from data. This has led to their increased application in computer vision tasks and image forensics [17]. Specifically, Vision Transformers are very efficient in capturing global and semantic data specific to the image. For instance, Chen et al., used Vision transformer along with a high frequency feature enhancement module to design a CG image detector which is robust to post-processing operations and adversarial attacks [18]. Darshan Lamichhane in his paper demonstrated the ability of Vision Transformer in capturing complex patterns in images, thus helping in AI image detection [19]. On the other hand, the pixel–level statistics carry minute discriminating information which may not be captured by these models. These can be obtained by handcrafted features.

The present paper combines the features of both in an attempt to enhance the performance and reliability of fake image detection model. This is inspired by the fact that, both global as well as local features of an image carry complementary information that are specific to the source of the image, which when combined can result in improved performance. This is achieved by using transfer learning from pre-trained models and combining this information with handcrafted and forensic information. The proposed model implements a multi-branch approach, where each branch extracts discriminating features. Cross-attention block is used to help the branches to learn from each other, and output enriched feature vectors. These feature embeddings are then fused using attention pooling, and then used for classification.

3. Proposed Method

The present work uses multi-branch and domain-adaptation approach to address the problem of distinguishing fake image from a real photograph. Each of the branches in the model extracts complimentary features, which are fused and then used for classification. The model uses two pre-trained Vision Transformer models – DINO [20] and CLIP [21] – in order to extract global information from the image. In addition, it uses forensic and handcrafted feature branches in order to extract primitive features. Features from each of the branches learn from the knowledge of the remaining branches with the help of cross attention. The information enriched embedding from the four branches is then fused using attention pooling, resulting in a final feature vector. This is then used for final classification. Augmentation invariant training is used to increase the reliability of the model under different real-world corruptions like noise, blur, compression, etc. The proposed model framework is given in fig. 2.

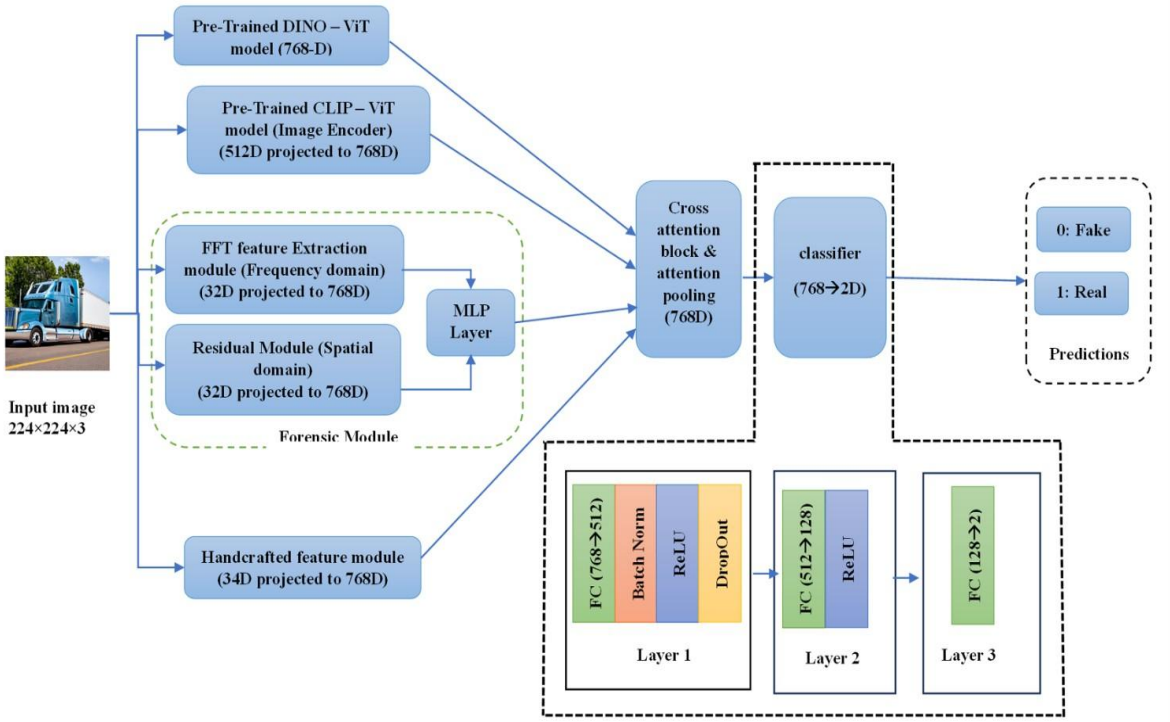


Fig. 2. Proposed Model Architecture.

Each of the block is described in detail below.

3.1. DINO – ViT Branch

DINO (self-**di**stillation with **no** labels) is a self-supervised Vision transformer (ViT) model, that was introduced by Meta AI[20]. It is pre-trained using teacher-student distillation framework on ImageNet – 1k dataset, which has 1.28 million real images divided into 1000 classes. It uses patches each of size 16 × 16. During pre-training, it creates multiple augmented views of an image like crop, color jitter, etc. The teacher and student models (both are Vision Transformers) train on each view. The student network is trained using knowledge distillation in such a way that it should match the output of teacher network. The parameters of the student network, θ_s are learnt in such a way that they minimize a loss function. On the other hand, the parameters of the teacher network are updated using an Exponential Moving Average on the student parameters as follows:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s \tag{1}$$

Where θ_t indicates the parameters (weights) of the teacher network, θ_s indicates parameters of the student network and $\lambda \in [0,1]$ is the momentum coefficient (usually close to 1, e.g., 0.996 -0.9995). In DINO training, the value of λ is increased slowly over around 300 epochs using cosine scheduling, for example, from 0.996 to 0.9995. This makes the teacher update more stably over time, making the student learn from a stable target. The DINO-ViT model architecture is shown in fig. 3.

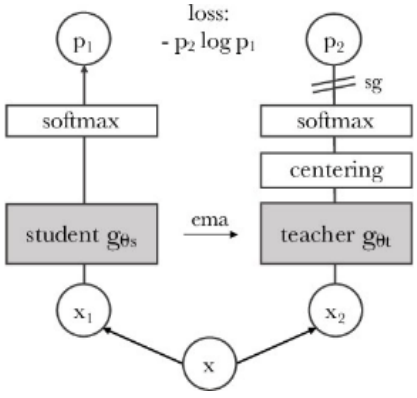


Fig. 3. DINO-ViT Architecture.

In each vision transformer (either student or teacher), the 224×224×3 RGB input image is split into 16 × 16 size patches. Each patch is then flattened and linearly projected. A special class token that is trainable is appended before the patch sequence for classification. In order to preserve the spatial information, a positional embedding is added to the patch embeddings. This sequence is then passed through a series of 12 transformer layers, where each layer performs the following major operations: multi-head self-attention (it helps in learning relation between image patches), MLP/feedforward neural network, and residual connections (for stability and gradient flow). The weights in each block decide what kind of information is emphasized by that block. The initial blocks emphasize general visual features and relation between them, like color, texture and local patch statistics. The later blocks, on the other hand, focus on global content of the image like object structure and composition of the scene.

In the proposed model, the DINO – ViT student model is instantiated. Most of its initial transformer blocks are kept frozen in order to retain most of its strong pre-trained information/features, and to avoid overfitting. But, the top 4 blocks are unfrozen in order to allow slight adaptation to subtle cues like texture smoothness, unnatural reflections, and artifacts.

The final output [CLS] token is a 768-D feature vector representing global image structure and composition.

3.2. CLIP – ViT branch

CLIP is also a Vision Transformer developed by the OpenAI and trained by Contrastive Language – Image Pretraining (CLIP) framework on OpenAI's dataset. This dataset has 400 million image-text pairs including real, graphics, art, etc. The text associated with each image is the natural language description of the corresponding image. By aligning the image and the corresponding descriptive text, the model learns visual representations of the images. It has an image encoder (vision transformer) and a text encoder.

During pre-training, each image is passed through the image encoder (ViT), which outputs an image embedding. The corresponding text is passed through the text encoder, which outputs a text embedding [21]. The model is trained to pull together the matching pairs, and push apart those which are not matching, using contrastive loss function.

In the image encoder part, the 224 × 224 × 3 RGB input image is split into 32 × 32 size patches. Each patch is then flattened and linearly projected. A class token that is trainable is appended before the patch sequence for classification. In order to preserve the spatial information, a positional embedding is added to the patch embeddings. This sequence is then passed through a series of 12 transformer layers, similar to that of DINO model.

The text encoder part is a transformer-based architecture, in which the input text is tokenized into subword units using a Byte Pair Encoding (BPE) tokenizer. Each token is embedded and added with a learnable positional embedding, in order to preserve order information. This is then passed through a series of 12-layer transformer block.

The final embedding of the end-of-text token is extracted, which is the representation of entire text. It is then passed through a linear projection layer to output a 512-D output feature vector.

The CLIP-ViT model framework is shown in fig. 4.

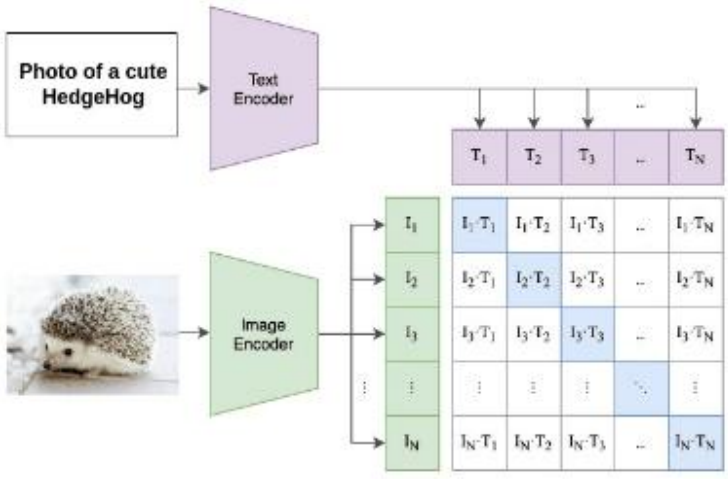


Fig. 4. CLIP-ViT model Architecture.

The text embedding and image embedding are normalized and compared using cosine similarity. During training, the objective is to get a maximum value for similarity between matching (image, text) pairs and minimum value for mismatched ones using a contrastive loss.

In the proposed model, the pre-trained image encoder of the CLIP-ViT [21] is instantiated. The last 2 blocks are unfrozen to allow slight adaptation. The output is a 512-D feature representing the context alignment and semantic consistency of the image. This is projected to 768-D to allow fusion with embedding from DINO.

3.3. Forensic Branch

This branch detects low-level forensic specific features like pixel-level anomalies, frequency anomalies and editing traces. It uses two complementary branches: FFT and residual branches.

3.3.1. FFT branch

This branch focuses on frequency domain information. It uses phase of FFT of the image in order to capture manipulation traces, compression artifacts, texture periodicity and regularity[22,23]. Since phase is more invariant to compression, phase instead of magnitude is considered in the model. Also, the graphic images have more regular transitions in phase, whereas real photos have more randomness in the phase transitions.

Since the phase is highly similar across color channels, the image is converted to grayscale to avoid redundancy. The 2D phase map of the grayscale image is fed to a CNN, which extracts a 32-D feature vector representing the image's structural irregularities. This is in turn projected to 768-D to allow for fusion with other branches of the model.

3.3.2. Residual Branch

This branch focuses on the information from spatial domain representation of the image. It captures the local noise inconsistencies, compression and pixel-level artifacts, etc. which provide information regarding sensor noise, demosaicing patterns, etc. These features are extracted by using Laplacian kernel for high pass filtering the image, resulting in edge and noise information. This information is helpful, since real photos have stochastic residual patterns, whereas the graphics have more uniform residual patterns[24].

The significance of phase and residual features have been validated with the help of entropy and visualization plots of five generators (SDV1.4, VQDM, GLIDE, ADM and BigGAN) and real images as shown in Fig. 5. The phase entropy plot shows that the real and diffusion model images all have higher and nearly identical entropy. However, the phase spectra visualizations show that real images have more heterogeneous patterns, whereas GANs and diffusion models have slightly regular patterns. The residual entropy plot on the other hand shows the stochasticity of the residual features, and the kurtosis plot shows the sharpness of the features. From the plots we can infer that real images have more random and smoother residual features reflecting the natural texture and sensor noise. The diffusion models have less stochastic and sharper residual features.

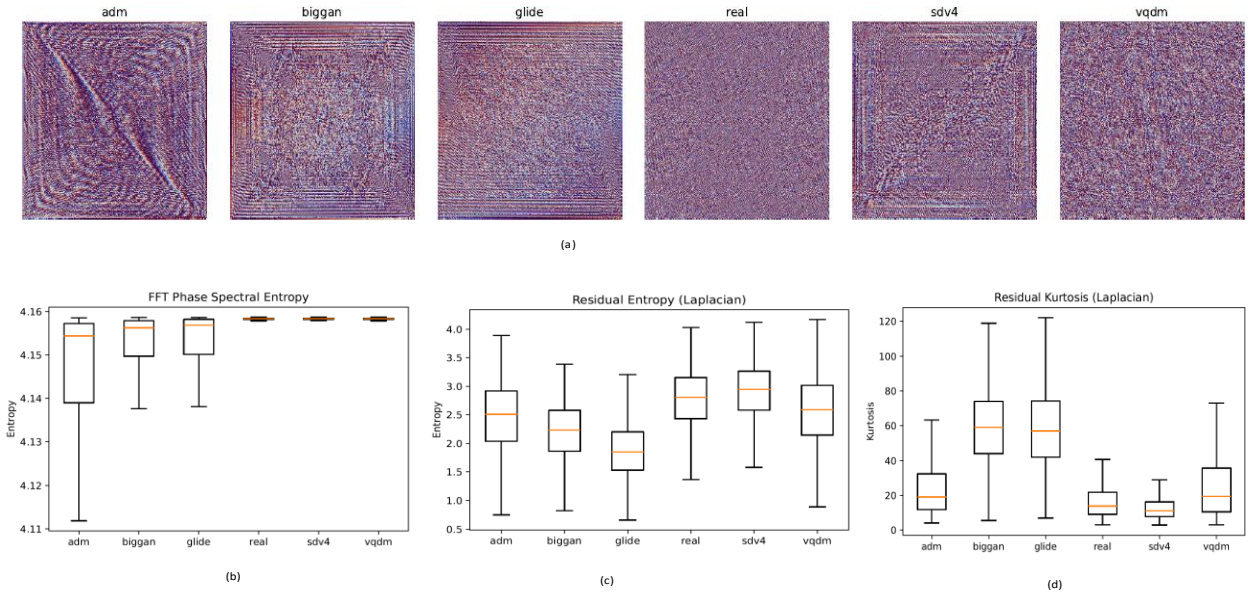


Fig. 5. (Phase spectra visualizations of fake and real images (b) entropy plot of phase spectra (c) entropy plot of residual features (d) Kurtosis plot of residual features.

In real photos, due to the image rendering process, there may be a statistical mismatch of edges/noise across the color channels. So, all the three channels of the image are convolved with Laplacian kernel separately. This is then fed to a CNN which results in a 32-D feature vector which is then projected to give a 768-D residual embedding.

A small MLP layer is used to learn a weighted concatenation of the embeddings from FFT and residual branches. The fused vector is the projected and normalized resulting in a 768-D forensic embedding, ready to interact with those from other three branches.

3.4. Handcrafted branch

This module extracts 34 low level statistical cues like color statistics, geometric structure and texture features. These features may be missed by the larger models (DINO and CLIP). This branch adds complementary information to the classification by modeling features like JPEG noise, inconsistencies in the gradient of the image, anomalies in the correlation between color channels and texture[8]. The 34 features belong to 3 groups: texture (15 features), shape (7 features) and color (12 features).

3.4.1. Texture features

The texture features include gradient, curvature, entropy, correlations and covariances of RGB channels, and textures of patches. The gradient features are the mean, variance, skewness and kurtosis of the gradient magnitude of the image obtained using Sobel operator.

$$\text{Sobel operator for gradient in x direction, } mask_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \tag{2}$$

$$\text{Sobel operator for gradient in y direction, } mask_y = mask_x^T \tag{3}$$

The gradient of the image I is obtained by convolving the image with above masks as follows:

$$\text{Gradient in x direction, } g_x = I * mask_x \tag{4}$$

$$\text{Gradient in y direction, } g_y = I * mask_y \tag{5}$$

$$\text{Gradient magnitude, } M_{grad}(x,y) = \sqrt{g_x(x,y)^2 + g_y(x,y)^2} \tag{6}$$

Mean, variance, skewness and kurtosis of the above gradient magnitude are obtained as below:
Let elem= number of elements

$$\text{Mean, } \mu = \frac{1}{elem} \sum_{x,\ y} M_{grad}(x,y) \tag{7}$$

$$\text{Variance, } \sigma^2 = \frac{1}{elem} \sum_{x,\ y} \left(M_{grad}(x,y) - \mu\right)^2 \tag{8}$$

$$\text{Skewness, } skew = \frac{\frac{1}{elem} \sum_{x,y} \left(M_{grad}(x,y) - \mu\right)^3}{\sigma^3} \quad \sigma = \text{standard deviation} \tag{9}$$

$$\text{Kurtosis, } kurt = \frac{\frac{1}{elem} \sum_{x,y} \left(M_{grad}(x,y) - \mu\right)^4}{\sigma^4} \quad \sigma = \text{standard deviation} \tag{10}$$

These values provide information like texture sharpness and edge density. The graphic images have more smooth gradient distributions compared to real images.

The curvature features are the mean and variance of curvature of the image calculated using second order derivative of the image, which are obtained by convolving the image with Sobel masks twice. This gives raw curvature components as follows:

$$\text{Curvature component in horizontal direction, } h_{xx} = g_x * mask_x \tag{11}$$

$$\text{Curvature component in vertical direction, } h_{yy} = g_y * mask_y \tag{12}$$

$$\text{Curvature component in diagonal direction, } h_{xy} = g_x * mask_y \tag{13}$$

These form the elements of the Hessian matrix, H, whose $H = \begin{bmatrix} h_{xx} & h_{xy} \\ h_{xy} & h_{yy} \end{bmatrix}$ eigenvalues give the principal curvatures along the directions of greatest and least curvatures at the particular pixel location. The trace and determinant of this Hessian matrix are used to calculate curvature descriptor at each pixel, as below

$$Curv(x,y)=\left(Tr(H)\right)^2\text{-}4*det(H) \tag{14}$$

The mean and variance of the above are used as features contributing to curvature information of the image.

$$M_{curv}=\frac{1}{elem}\sum_{x,y} curv(x,y) \tag{15}$$

$$\Sigma^2_{curv}=\frac{1}{elem}\sum_{x,y}\left(curv(x,y)\text{-}\mu_{curv}\right)^2 \tag{16}$$

These help in determining structural complexity. Fake images have simpler curvature patterns than real images.

Shannon entropy of the image tells the randomness of pixel intensities. The normalized histogram values for the image are calculatedp_i over 32 bins (b=32), and entropy is calculated as follows:

$$H(I)=\text{-}\sum_{i=1}^{b} p_i\, log\left(p_1\right) \tag{17}$$

Natural images tend to have higher entropy owing to sensor noise or complex texture.

In order to obtain the correlation and covariance values between the color channels of the image, its Red, Green and Blue channels are separated, and the corresponding values are obtained as follows:

Let R, G, B = separated color channel matrices of the image I. then covariance between R and G channels,

$$Cov_{RG}=mean(R*G)\text{-}mean(R)*mean(G) \tag{18}$$

Similarly, we obtain covariance between the other pairs, $cov_{GB}\ and\ cov_{RB}$

The correlation coefficients between each pair of channels are obtained as follows:

$$P_{RG}=\frac{cov_{RG}}{\sigma_R*\sigma_G} \tag{19}$$

where $\sigma_R$and $\sigma_G$are the standard deviations of R and G channel matrices respectively

Similarly, the correlation coefficient values for the other pairs are obtained, as $\boldsymbol{\rho_{GB}}\boldsymbol{and}\ \boldsymbol{\rho_{RB}}$

Computer generated images have high correlations and covariances between the RGB color channels due to the synthetic generation process.

Graphics have unnaturally uniform/smoother textures, and the mean and variance of local patches help in determining the local texture smoothness. These are calculated as follows: The image is divided into 3 × 3 patches I_{patch}, and the mean and variance of each patch is calculated. The variance map gives the local contrast. This is then summarized into two values, mean and standard deviation of the variance map.

$$Mean_{patch}=\frac{1}{9}\sum_{x,y} I_{patch}(x,y) \tag{20}$$

$$Variance\ map,\ var_{patch}(x,y)=\frac{1}{9}\sum_{x,y}\left(I_{patch}(x,y)\text{-}mean_{patch}\right)^2 \tag{21}$$

$$Var\ map_{mean}=mean\ \left(var_{patch}(x,y)\right) \tag{22}$$

$$Var\ map_{std}=\sqrt{mean\left[\left(var_{patch}(x,y)\text{-}var\ map_{mean}\right)^2\right]} \tag{23}$$

So, the 15 texture features are obtained from (7) to (10), (15) to (19) and (22) to (23).

$$Txr_{ftr}=[\mu,\ \sigma^2,\ skew,\ kurt,\ \mu_{curv},\ \sigma^2_{curv},\ H(I),\ \rho_{RG},\ \rho_{GB},\ \rho_{RB},\ cov_{RG},\ cov_{GB},\ cov_{RB},\ var\ map_{mean},\ var\ map_{std}] \quad (24)$$

3.4.2. Shape features

The shape features are the 7 Hu Moments. These are insensitive to geometrical variations of the image like scaling rotation and translation. They are shape descriptors that help detect any structural variations or geometrical inconsistencies. In order to summarize them, we first obtain the raw geometrical moments using the following equations:

$$M_{pq}=\sum_x\sum_y x^p y^q I(x,y) \quad (25)$$

The zeroth order moment (p=q=0) give the total intensity of the image, whereas the first order moments (p=0, q=1 or vice versa) give the centroid, which is obtained as follows:

$$\bar{X}=\frac{m_{10}}{m_{00}},\ \bar{y}=\frac{m_{01}}{m_{00}} \quad (26)$$

To make the features invariant to translation and scaling, we use normalized central moments:

$$central\ moments,\ \mu_{pq}=\sum_x\sum_y (x-\bar{x})^p y(y-\bar{y})^q I(x,y) \quad (27)$$

$$Normalized\ central\ moments,\ \eta_{pq}=\frac{\mu_{pq}}{\mu_{00}^{1+\frac{p+q}{2}}} \quad (28)$$

In order to further make these features rotation invariant, we use Hu's seven invariant moments which are the polynomial combinations of η_{pq} and are represented as $Hu=[\varphi_1,\varphi_2,\cdots,\varphi_7]$. [25]. In order to ensure that these values lie in the same manageable range, these are stabilized using log transformation.

$$\Phi_f^i=sign(\varphi_i)*log(1+|\varphi_i|) \quad (29)$$

So, the 7 shape features are

$$Shape_{ftr}=\left[\varphi_f^1,\ \varphi_f^2,\ \varphi_f^3,\ \varphi_f^4,\ \varphi_f^5,\ \varphi_f^6,\ \varphi_f^7\right] \quad (30)$$

3.4.3. Color features

HSV representation captures intensity and saturation variations. These features help in improving the reliability of the model under any color manipulations. In addition, Lab representation captures perceptual color uniformity and models chrominance distortions quite well. This helps in making the model robust to JPEG compression, since compression results in distortions in chrominance. This information is modeled using the mean and standard deviation of each channel in the HSV and Lab representations of the image.

$$Mean,\ \mu_{Chnl}=\frac{1}{elem}\sum_{x,y} chnl(x,y) \quad (31)$$

$$Standard\ deviation,\sigma_{chnl}=\sqrt{\frac{1}{elem}\sum_{x,y}\left(chnl(x,y)-\mu_{chnl}\right)^2} \quad (32)$$

Where $chnl = H\ or S\ or\ V\ or\ L\ or\ a\ or\ b\ for\ each\ color\ space$
So, the 12 color features are

$$Color_{ftr}=[\mu_H,\sigma_H,\mu_S,\sigma_S,\mu_V,\sigma_V,\mu_L,\sigma_L,\mu_a,\sigma_a,\mu_b,\sigma_b] \quad (33)$$

The 34-D handcrafted feature is obtained by concatenating the texture, shape and color features from (24), (30) and (33).

$$Hand_{vector}=[txt_{ftr}, shape_{ftr}, color_{ftr}] \tag{34}$$

This vector is then projected to 768-D in line with the other three branches.

3.5. Cross-attention block and attention pooling

The object-level semantic features from the DINO block, the semantic contextual features from the CLIP block, the manipulation-sensitive features from the forensic block, and the low-level statistical features from the Handcrafted feature block are stacked by the cross-attention module. The embeddings from each branch are updated with information from other branches. These information-enriched embeddings are then fused via attention pooling resulting in a final 768-D fused embedding. This is then fed to a final prediction head and a projection head. The mathematical formulation for the fusion mechanism is given below.

Let N = the number of branches,
H=dimension of feature embedding of each branch
P=number of heads
d_h = H/P
The feature embedding of each branch can be denoted by

$$f_i \in \mathbb{R}^H, i = 1, 2, \dots, N \tag{35}$$

These feature embeddings are stacked to form a sequence,

$$F = [f_1, f_2, \dots, f_N] \in \mathbb{R}^{B \times N \times H} \tag{36}$$

Multi-head self-attention is applied across all the branches with query, key and value projections learnt with F as token as follows:

$$Q_h = FW_Q^{(h)}, K_h = FW_K^{(h)}, V_h = FW_V^{(h)} \tag{37}$$

Where $W_Q^{(h)}, W_K^{(h)}$ and $W_V^{(h)} \in \mathbb{R}^{H \times d_h}$ are learnt projection matrices.
For each head h, the attention is computed as

$$Attn_h = softmax\left(\frac{Q_h {K_h}^T}{\sqrt{d_h}}\right)V_h \tag{38}$$

These attentions from all the heads are concatenated and projected to give

$$\hat{F} = concat\ (Attn_1, \dots, Attn_P)W_o \in \mathbb{R}^{B \times N \times H} \tag{39}$$

Where W_o is the output projection.
In order to stabilize and preserve the original feature information, residual connection is applied followed by normalization

$$\tilde{F} = LayerNorm\left(\hat{F} + F\right) \in \mathbb{R}^{B \times N \times H} \tag{40}$$

This is then given to an adaptive attention pooling layer, which learns weights for each branch and aggregates it into a single fused feature vector.
The weights for each branch in attention pooling are learnt as

$$\alpha_i = \frac{\exp\left(w^T \tilde{f}_i\right)}{\sum_j \exp\left(w^T \tilde{f}_j\right)} \tag{41}$$

The final fused vector output is

$$f_{fused} = \sum_{i=1}^{N} \alpha_i \tilde{f}_i \in \mathbb{R}^{B \times H} \tag{42}$$

For the proposed model,
The number of branches, N=4
Dimension of feature embedding of each branch, H=768
The number of heads used, P=8
The batch size, B=32

3.6. Classifier

The prediction head classifies the feature resulting in output logits for two classes (real and fake). It is an MLP head containing three layers. The first layer has linear layer, BatchNorm, ReLU and DropOut layers. This layer projects the embeddings from 768-D to 512-D. The second layer contains linear and ReLU layers. This projects from 512-D to 128-D. The last layer has linear layer projecting to 2-D (logits).

3.7. Contrastive learning and Loss functions

The projection head, on the other hand, is an MLP layer that projects the feature embedding from 768-D to 256-D, and then to 128-D required for contrastive learning. The contrastive learning trains the model in such a way that similar embeddings are pulled together, whereas different ones are pushed apart.

In our model, this is achieved by using two contrastive learning losses: the supervised contrastive loss that ensures both inter class separability and intra-class compactness, and the augmentation-invariant loss, which ensures that different augmented views of the same image are close to each other. In addition to these two loss functions, we also use center loss that ensures that the embeddings of same class are close to the mean embedding of that class. This further enhances intra class compactness. Further, we use cross entropy loss on each branch, and with the final classifier. Each of the FFT and residual branches of the forensic module and the handcrafted branch has an auxiliary classifier, which learns to classify the image with its embedding alone with the help of cross entropy loss. This helps in strengthening the individual branches.

4. Experimental Results

4.1. Datasets

The proposed model uses GenImage[26], CIFAKE[27] and PRCG[28] datasets. The GenImage dataset contains 1,331,167 real and 1,350,000 fake images on par with ImageNet dataset. It has 1000 classes, and 1350 generated images in each class[26]. The original dataset employs eight generative models for image generation, namely BigGAN [1], GLIDE [29], VQDM [30], Stable Diffusion v1.4 and v1.5 [31], Midjourney, ADM [32] and Wukong. The Photo - Realistic Computer Generated (PRCG) images include categories of architecture, nature, object, life and game. They are mainly collected from 40 3D-graphics websites. The set contains 800 PRCG images generated using rendering softwares like 3ds MAX, softimage-xsi, Maya, Terragen, and so on. The geometry modeling tools include AutoCAD, Rhinoceros, softimage-3D, and so on. The dataset also contains 800 natural photographic images (PIM) images which are obtained from Google, searched with keywords that match the categories within the PRCG set[28]. The PRCG images are stored in .JPG format and their size varies between 400 pixels and 1400 pixels in average dimension, (width + height)/2. The PIM are also stored in .JPG format, and their size varies between 315 pixels and 1500 pixels in average dimension, (width + height)/2. The CIFAKE dataset has 60,000 real images collected from CIFAR-10 dataset [27]. They are divided into 10 classes like airplane, automobile, bird, etc. with 6000 images per class. It has 60,000 fake images equivalent to the CIFAR-10 images, generated using stable diffusion model.

GenImage dataset has eight generators. Each generator has 168,000 real and 168,000 fake images. The images in each generator were split into train, validation and test sets in the ratio 80:10:10. In order to evaluate the model on this dataset, the model was trained using images from SDV1.4 in line with the other baselines used for comparison. Similarly, for the second dataset, the images are divided into train, validation and test sets in the ratio 80:10:10. The CIFAKE dataset is composed of 50K training images and 10K test images, each of real and fake images. The training set is split into training and validation folders in the ratio 80:20. In addition due to computational constraints, to analyze the cross-generator performance of the model, we have evaluated on the images from the five generators of GenImage (SDV1.4, VQDM, ADM, GLIDE and BigGAN).

All the images are resized to 224 × 224 and normalized by a mean and standard deviation consistent with the pre-trained CLIP encoder.

4.2. Experimental Settings

4.2.1. Training

The proposed model uses a size of 224 × 224 for all the images, and normalizes them to have a mean and standard deviation matching with the pre-trained CLIP model. In order to improve the robustness of the model to the real-world distortions like noise, blur, compression, occlusion, color variations among others, augmentation-invariant training is employed. For each image in the training set, two random augmented views are generated. During training, the embeddings of different augmentations of same image are pushed closer together. Instead of learning from a single loss term, this model uses multi-objective training, wherein, in addition to learning the model weights, learning also happens in each branch. In addition, a distance-based regularizer is used on the final embeddings to ensure more intra-class compactness. This is done by learning a center for each class, which gets updated along with the model weights in such a way that, the features of a particular class move closer to the corresponding center.

We have used NVIDIA Tesla T4 GPU for model training. The Optimizer used is AdamW optimizer, number of training epochs are 15, batch size is 32, and learning rate (LR) scheduler is Cosine Annealing LR scheduler for the first 70% of epochs, and stochastic weighting average (SWA) scheduler for the last 30% of the epochs, in order to ensure a smoother model more robust to noise. The DINO and CLIP branches use a very small initial LR (5e-6) in order to avoid large modification in the weights and catastrophic forgetting of the already pre-trained knowledge. The remaining two branches, the fusion block and the final classifier use a moderate LR (1e-4) since these are trained from scratch. The LR for the class center for the distance-based regularizer is high (0.5) in order to ensure faster clustering of the feature embeddings.

In order to ensure statistical robustness of the model, the experiment was repeated three times with three different random seeds (seed = 43, 123, 769) controlling data sampling and model initialization, and the results are presented as mean ± standard deviation over the three independent runs. Due to computational constraints, this was restricted to CIFAKE and PRCG dataset. Also, all the ablation experiments were conducted on CIFAKE dataset with a fixed random seed (seed=43).

4.2.2. Evaluation

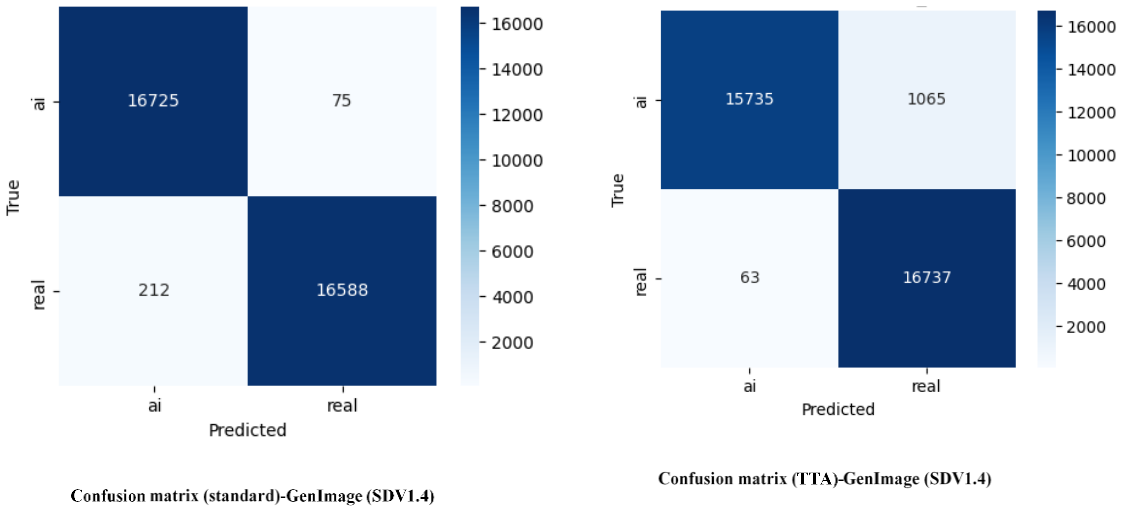
In order to ensure scale-invariance, and to avoid bias towards a particular scale, multi-scale inference is used, where each test image is evaluated at three different scales, and the predictions are averaged. This is the standard evaluation. In addition, Testing Time Augmentation (TTA) evaluation is also done in order to test the reliability of the model under real-world corruptions. Each test image is augmented five times randomly, and the predictions are averaged. The augmentations include color variations, compression, camera noise/sensor noise, blur, occlusion, illumination variation and others.

We evaluated our model on three publicly available datasets: CIFAKE (~60K each of real and fake images), PRCG (~800 real and 800 fake images) and GenImage (~168K real and 168K fake images) datasets. The performance of the model is compared with other baselines which used the above datasets. Further, the performance of the model under different perturbations was evaluated to analyze its robustness. The performance of the model is analyzed with the help of evaluation metrics –accuracy, precision, recall, F1 score, confusion matrix, AUC, and t-SNE plot. In addition, the performance of the model over unseen generators was measured using GenImage dataset.

4.3. Performance analysis

4.3.1. Performance of the model on individual datasets

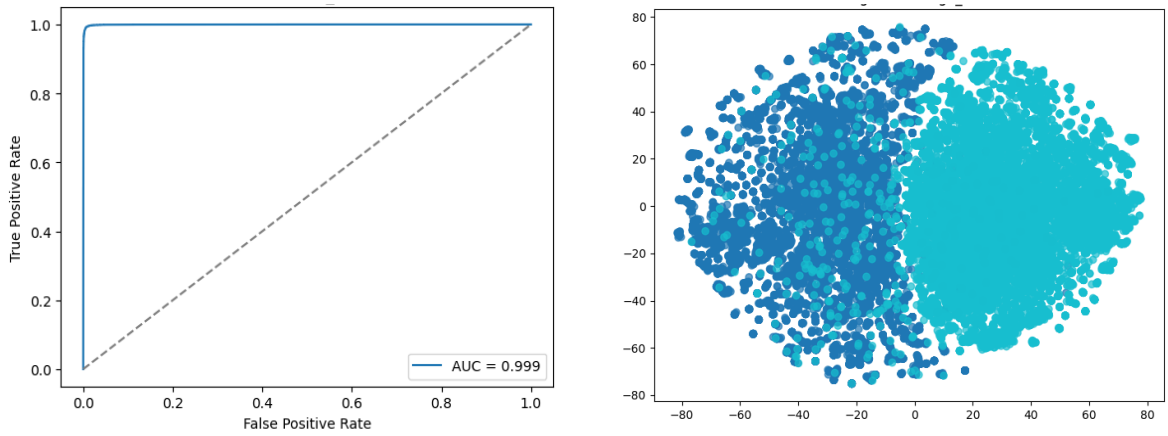
For the GenImage dataset, we trained and tested on images from SDV1.4 generator, and we observed an overall classification accuracy of 99.15% when raw test images were used. In addition, the model was also evaluated by subjecting each test image to 5 random augmentations like illumination or color variation, noise, blur, occlusion, and averaging the predictions. This test-time augmentation (TTA) evaluation gave a classification accuracy of 96.64%. The confusion matrices for both standard and TTA evaluation are shown in Fig. 6. From the confusion matrix, it can be seen that for the raw test images, the model correctly classifies 16725 out of 16800 fake images, and 16588 out of 16800 real images.



(a) (b)

Fig. 6. Confusion Matrix on GenImage (SDV 1.4) dataset for (a) standard evaluation (b) TTA-based evaluation

The ROC curve is in Fig. 7(a). The Area Under Curve is 0.999, which indicates that our model has good discriminating ability. Also, the t-SNE plot is shown Fig. 7(b) which shows good intra-class compactness of the model on the dataset. However, there is a slight overlap between the images of the two classes. When a few misclassified samples were observed, the overlap mostly corresponded to images which are visually ambiguous like high quality photorealistic AI rendered images, real-life scenes, etc. A few misclassified images are shown in Fig. 8.



(a) (b)

Fig. 7. (a) ROC Curve (b) t-SNE plot of the model for Genimage (SDV1.4) dataset.

Fig. 8. Misclassified samples from SDV1.4 dataset.

The other performance metrics (Precision, Recall and F1-Score) for this dataset are shown in table 1.

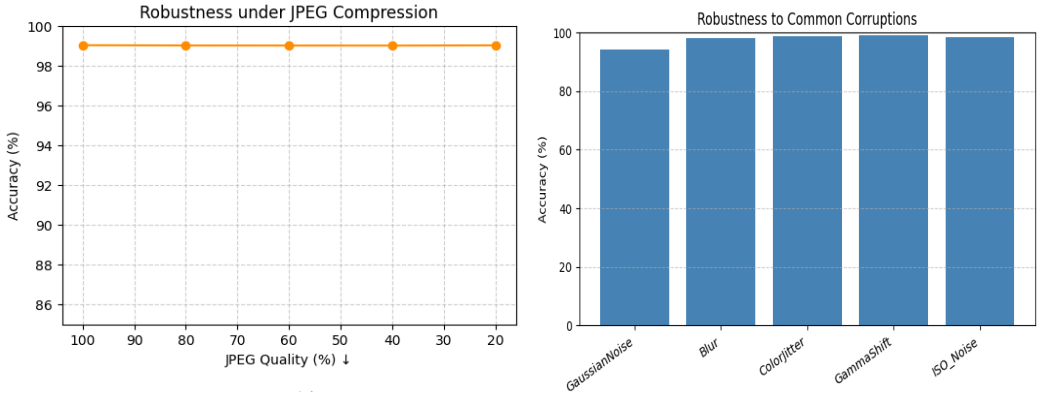
In addition to the above evaluation, the model was also tested under different perturbations. It was tested for performance when the test images were JPEG compressed with different Quality factors ranging from 100 (low compression) to 20 (high compression). And the model showed almost stable accuracy as shown in Fig. 9(a). However, the datasets contain JPEG-encoded images, resulting in inherent compression in the training data before augmentation. So, most of the stability can be attributed to the data distribution matching, and very mildly to the augmentation

invariant training. This was confirmed by evaluating the model with augmentation disabled. This did not cause collapse in robustness to compression, and showed a very mild increase (~0.2%) in accuracy when augmentation was enabled.

Table 1. Performance metrics for the model on GenImage (SDV1.4) dataset.

Class	Precision	Recall	F1 Score	Support
Fake	0.9955	0.9874	0.9914	16800
Real	0.9875	0.9955	0.9915	16800
Average	0.9915	0.9915	0.9915	33600 (Total)

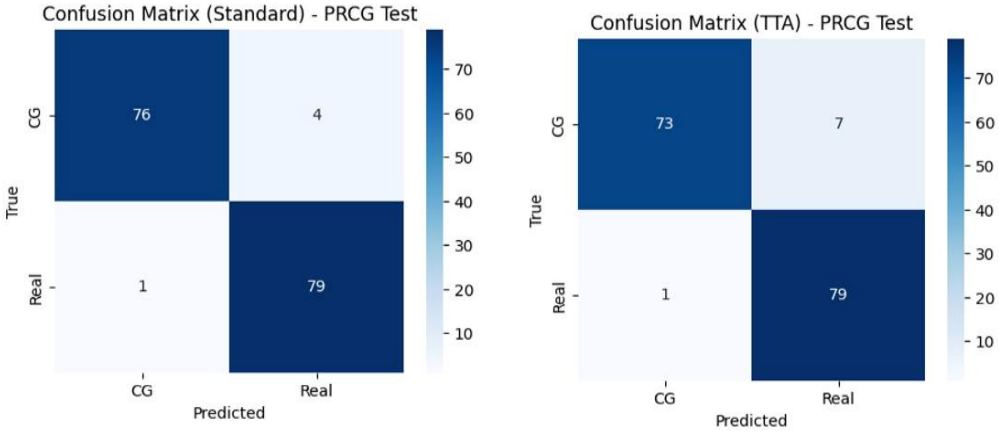
The other degradations tested are as follows: Gaussian Noise with zero mean and variance in the range 20 to 50, Gaussian blur generated with a kernel of maximum size 5×5, contrast/saturation/hue/brightness variation, camera/sensor noise, and, tone/non-linear brightness adjustment using Gamma correction with gamma value ranging between 0.6 to 1.4. The accuracy obtained when the test images were subjected to each of the above perturbations is shown in Fig. 9(b). The Mean Robust Accuracy over all the perturbations obtained is 98.36%.



(a) (b)

Fig. 9. Accuracy of the model for GenImage(SDV1.4) dataset (a) when images are JPEG Compressed with different Quality factors (b) for different perturbations.

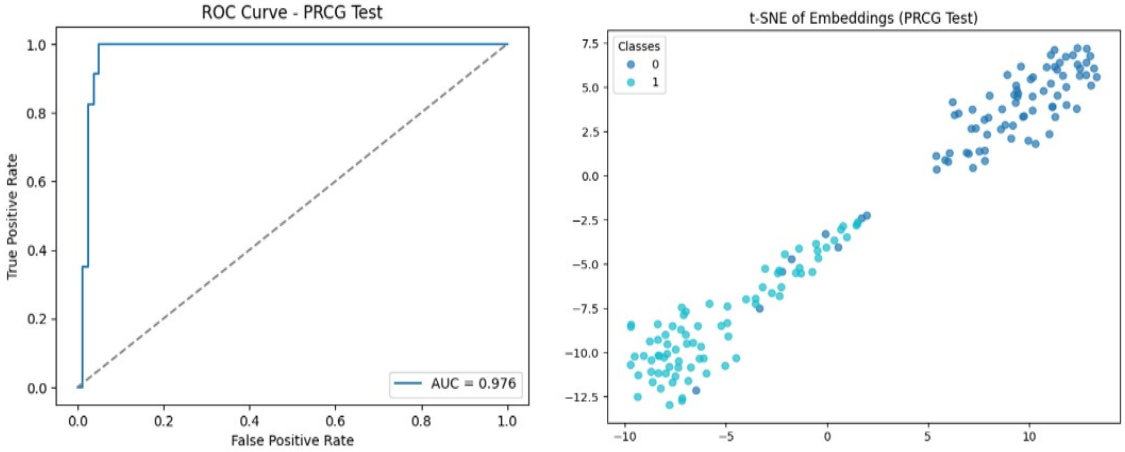
Similar experiments were done for PRCG dataset. The experiment was repeated with three random seeds (seeds=43, 123 and 769), and the results were averaged. The raw test images gave an average classification accuracy of 95.83±0.36% whereas the test-time augmentations displayed a classification accuracy of 94.83±0.87%. The confusion matrices for a sample seed are shown in Fig. 10 below.



(a) (b)

Fig. 10. Confusion Matrix for (PRCG Dataset) for (a) standard Evaluation (b) TTA-based evaluation

Area under the ROC curve for the PRCG dataset gave an average of 0.98 ±0.003 over the three seeds, and the t-SNE plot is shown in figure a displaying clear separability between the classes for a sample seed as shown in fig. 11.



(a) (b)

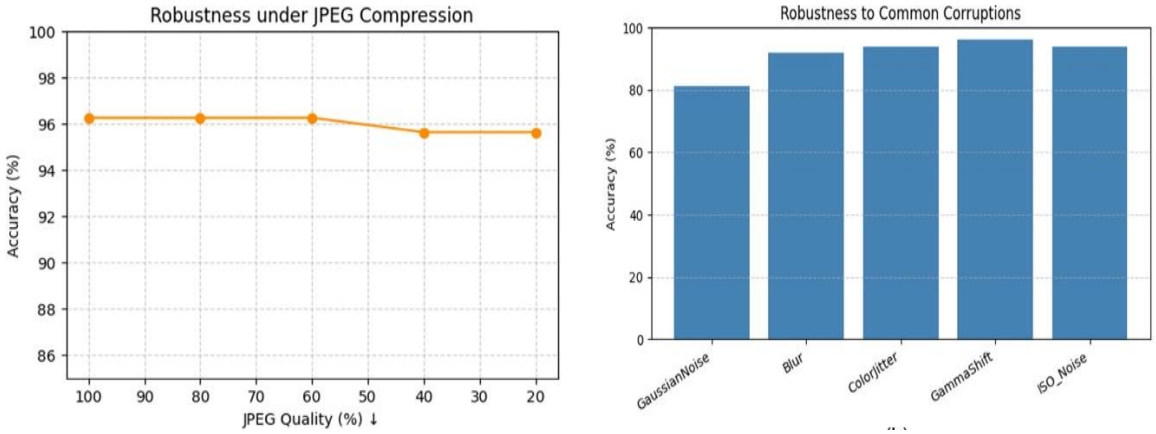
Fig. 11. (a) ROC Curve (b) t-SNE plot of the model for the PRCG dataset.

The other performance metrics (Precision, Recall and F1-Score) for this dataset are shown in Table 2.

Table 2. Performance metrics for the model with seed=43 on the PRCG dataset.

Class	Precision	Recall	F1 Score	Support
Fake	0.9870	0.9500	0.9682	80
Real	0.9518	0.9875	0.9693	80
Average	0.9694	0.9688	0.9687	160 (Total)

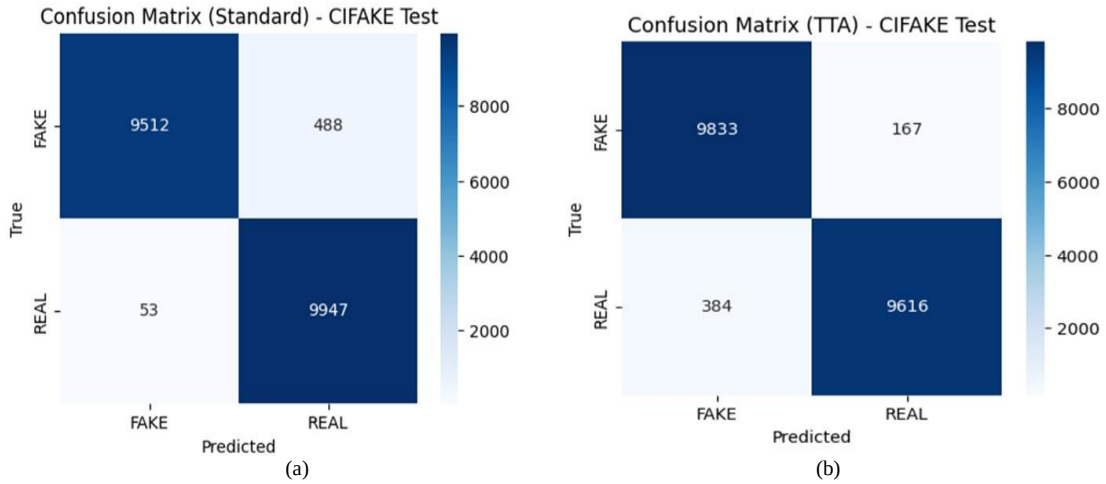
The performance of the model under JPEG compression with different quality factors is shown in fig. 12(a). The performance for other perturbations for the PRCG dataset is shown in fig. 12(b). The Mean Robust Accuracy over all the perturbations for this dataset is 93.69%.



(a) (b)

Fig. 12. Accuracy of the model for PRCG dataset (a) when images are JPEG Compressed with different Quality factors (b) for different perturbations

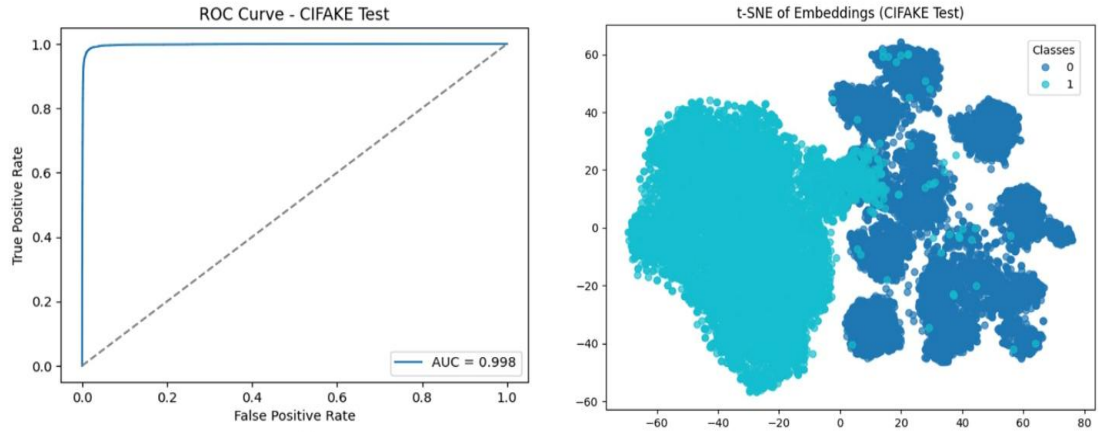
With CIFAKE dataset, the model gave an average test accuracy of 96.79 ±0.17% over the three seeds with raw test images, whereas it displayed 95.57 ±0.82 % accuracy with images subjected to test-time augmentations. The confusion matrices in the two cases are in fig. 13.



(a) (b)

Fig. 13. Confusion Matrix for the CIFAKE Dataset for (a) standard evaluation (b) TTA-based evaluation.

The area under the ROC curve obtained for this dataset is 0.997 ± 0.001. The ROC curve and the t-SNE plot are shown in fig.14 for sample seed=43.



(a) (b)

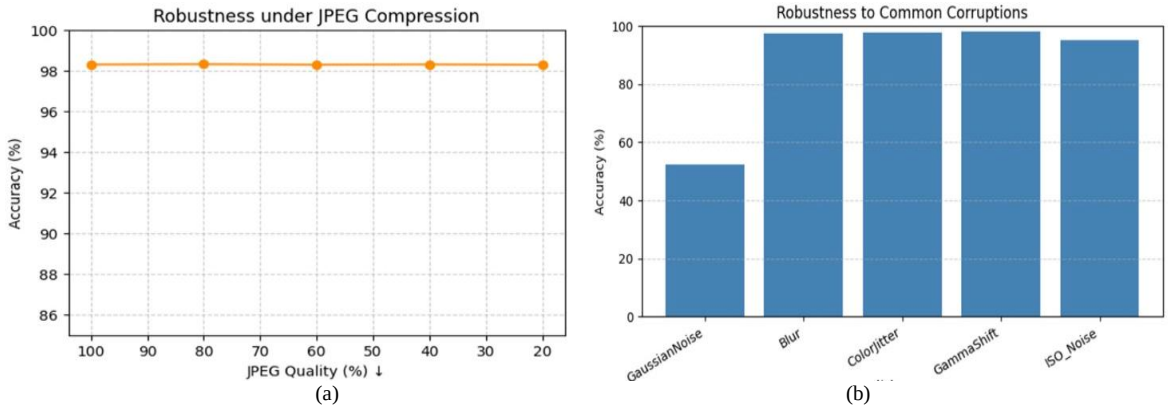
Fig. 14. (a) ROC Curve (b) t-SNE plot of the model for the CIFAKE dataset.

The other performance metrics (Precision, Recall and F1-Score) for this dataset are shown in table 3 for a sample seed=43.

Table 3. Performance metrics for the model on CIFAKE dataset.

Class	Precision	Recall	F1 Score	Support
Fake	0.9945	0.9512	0.9723	10000
Real	0.9532	0.9947	0.9735	10000
Average	0.9738	0.9729	0.9729	20000

The performance of the model under different levels of JPEG compressions are displayed in fig. 15(a). Also, the response under different types of perturbations for this dataset are shown in fig. 15(b). This dataset gave a mean robust accuracy of 93.21%.



(a) (b)

Fig. 15. Accuracy of the model for CIFAKE dataset (a) when images are JPEG Compressed with different Quality factors (b) for different perturbations.

Comparison of model performance with other baselines for each dataset is shown in table 4. The results show that our proposed model gave an enhanced performance on all CIFAKE and PRCG datasets and comparable performance in case of GenImage (SDV1.4) dataset as compared to other baselines.

Table 4. Performance comparison (average Accuracy) of the proposed model with other Baselines for the PRCG and CIFAKE datasets over 3 seed (seed=43,123, 769). Accuracy for GenImage (SDV1.4) dataset was evaluated for a single seed due to computational constraints.

Dataset	Model	Total Accuracy
PRCG dataset	Manjary P Gangan et al[10]	89.38%
	Chanthini Baskar et al[5]	90.6%
	Proposed Model (raw images)	95.83 ±0.36%
	Proposed Model (Test-time augmentations)	94.83 ±0.87%
CIFAKE dataset	Jordan J. Bird and Ahmad Lotfi[27]	92.98%
	Md Tanvir Islam et al[33]	94%
	Residual Networks (ResNet)[34]	91%
	Variational Autoencoders (VAEs)[34]	69%
	Proposed Model (raw images)	96.79 ±0.17%
	Proposed Model (Test-time augmentations)	95.57 ±0.82 %
GenImage dataset	Mingjian Zhu et al[35]	96.1%
	Jiaxuan Chen et al[36]	99.2%
	Proposed Model (raw images)	99.15%
	Proposed Model (Test-time augmentations)	96.64%.

4.3.2. Cross-generator performance with the GenImage dataset

We evaluated our proposed model when it is tested with images from unseen generators. For this purpose, we have used 5 generators of the GenImage dataset (SDV1.4, ADM, VQDM, GLIDE and BigGAN). Each generator has 168,000 real and 168,000 fake images, which are split into train, validation and test sets in the ratio 80:10:10. The model was trained on the training subset of each generator, and tested on the testing subsets of all other generators. We observed good performance when the training and testing was done on same generators, but degraded performance when the same process was done with different generators. We have compared our model with ResNet-50 model performance as analyzed by Zhu et al [19]. The comparison is shown in fig.16. The degradation in the performance may

Training Subset \ Testing Subset	Sdv4	Glide	BigGAN	Vqdm	ADM
Sdv4	99.9	61.9	52	56.6	53.5
Glide	50	99.9	74	51	56
BigGAN	49.9	68.4	99.9	50.6	50.6
Vqdm	50	60.1	66.8	99.9	50.7
ADM	53.1	97.1	88.3	61.5	99

(a)

Training Subset \ Testing Subset	Sdv4	Glide	BigGAN	Vqdm	ADM
Sdv4	99.2	60.52	53.09	55.45	55.6
Glide	53.16	99.82	88.98	63.35	58.3
BigGAN	49.98	74.39	99.9	54.85	52.8
Vqdm	55.94	72.22	77.31	99.35	60.73
ADM	49.99	94.02	87.01	62.3	98.9

(b)

Fig. 16. Cross-Generator performance of detectors when trained and tested on raw images from GenImage dataset using (a) ResNet-50[26] (b)proposed model.

be attributed to the forensic branch, which learns FFT phase and residual features. From fig.5, it can be observed that the phase and residual patterns are different for different generators. More specifically, the phase and residual entropies of images from SDV1.4 and ADM are more close to those of real images resulting in poorer performance for these generators as can be seen in fig.16. This indicates the model being biased by the generator-specific features resulting in across-generator performance degradation. In addition, as analyzed by Grommelt et al[12], the cross-generator performance also depends on compression and size bias of training set.

4.4. Ablation Results

All the ablations were conducted on the CIFAKE dataset and for single seed (seed = 43) in order to ensure computational feasibility.

4.4.1. Ablation results for multibranch architecture

In order to validate and justify the architectural complexity of the proposed model, we evaluated it with individual branches, and observed the change in performance parameters when each component is added to the model. Due to computational constraints, and the vast number of training images of the GenImage dataset (>300K), the ablation study was restricted to CIFAKE dataset. Table 5 shows the results of this ablation study. The branch ablation was conducted with cross-attention fusion. From the accuracy column, we can observe that, the DINO branch contributes maximum to the performance. The forensic and handcrafted branches are insufficient individually. Although, they are not dominant, they function as auxiliary features. In order to get maximum contribution from these branches, the fusion strategy plays an important role. The last two rows of the table show the contribution of cross-attention fusion of the branches, towards the performance of the model as compared to simple concatenation. It can be observed that cross-attention contributes around 1% improvement in accuracy compared to concatenation. Moreover, cross-attention fusion results in a drop in memory consumption by around 28% as compared to concatenation. This can be attributed to the compact branch-token representation of cross-attention.

Table 5. Ablation results of the multi-branch architecture and the contribution of the fusion strategy (CIFAKE dataset).

Model Variant	Params (M)	FLOPs (G) (per forward pass)	Inference peak memory	Training peak memory	Avg inference time	Std Accuracy (%)	TTA Accuracy (%)	per-component parameter count (M)
DINO only	33.77 M	12.02 GMAC	769.5 MB	793.0 MB	5.99 ms / image	95.65	95.61	DINO branch: 28.35 M Classifier head: 0.46 M
CLIP only	157.09 M	4.41 GMac	1163.4 MB	1169.5 MB	8.22 ms / image	94.82	94.48	CLIP branch: 151.28 M Classifier head: 0.46 M
Forensic only	6.66 M	40.26 MMAC	845.3 MB	874.2 MB	1.41 ms / image	74.58	69.46	Forensic branch: 1.25 M Classifier head: 0.46 M
Handcrafted only	5.62 M	1.0 MMac	858.1 MB	867.3 MB	14.30 ms / image	50.64	50.6	Handcrafted branch: 0.21M Classifier head: 0.46 M
DINO+ CLIP	185.44 M	16.44 GMac	2375.7 MB	2391.6 MB	16.39 ms / image	96.52	96.47	DINO branch: 28.35 M CLIP branch: 151.28 M Fusion module: 2.36 M Classifier head: 0.46 M
DINO+ CLIP+ FORENSIC	186.69 M	16.48 GMac	3126.0 MB	3153.9 MB	16.97 ms / image	96.66	96.48	DINO branch: 28.35 M CLIP branch: 151.28 M Forensic branch: 1.25 M Fusion module: 2.36 M Classifier head: 0.46 M
Complete Model (Fusion Strategy: Cross-Attention)	186.89 M	16.48 GMac	3845.3 MB	3873.0 MB	30.60 ms / image	97.43	97.31	DINO branch: 28.35 M CLIP branch: 151.28 M Forensic branch: 1.25 M Handcrafted branch: 0.21 M Fusion module: 2.36 M Classifier head: 0.46 M
Complete Model (Fusion Strategy: Simple Concatenation)	184.53 M	16.47 GMac	5341.2 MB	5369.3 MB	29.00 ms / image	96.46	96.39	DINO branch: 28.35 M CLIP branch: 151.28 M Forensic branch: 1.25 M Handcrafted branch:0.21 M Fusion module: 2.36 M Classifier head: 0.46 M

4.4.2. Ablation results for handcrafted feature contribution

In order to validate the significance of each of the three sets of handcrafted features (texture, shape and color features), an ablation study of these features was conducted on CIFAKE dataset. In addition, the discriminative power of each feature was measured with the help of mutual information using (43) and the results are tabulated in table 6.

$$MI(X;Y) = H(Y) - H(Y|X) \tag{43}$$

where

X=feature (texture/shape/color)
Y=ground-truth label
H(.)=entropy

The last column shows that the texture features contribute maximum discriminative power to the model, whereas the shape features contribute negligibly. This is further confirmed from the accuracy column of the table that omitting the shape features would have negligible effect on the accuracy of the model. This can be attributed to the fact that most of the geometrical features are preserved by most generative models.

Table 6. Ablation results of handcrafted features for the CIFAKE dataset.

Handcrafted features	Accuracy (%)	average MI
texture only	96.93	0.0146
shape only	95.94	0.0004
color only	96.35	0.0058

4.4.3. Ablation results for contrastive learning loss functions

While calculating the final loss during contrastive learning, the weights for the loss functions were selected ensuring that the cross-entropy objective is dominant. So it was given full weightage ($\lambda_{ce} = 1$). Since they contribute to robustness, the supervised contrastive loss and augmentation-invariant loss functions were given medium weightage ($\lambda_{con} = \lambda_{aug} = 0.1$). The auxiliary losses of the forensic and handcrafted branches were assigned lower weightage ($\lambda_{fft} = \lambda_{res} = \lambda_{hand} = 0.05$). Finally the center loss, in order to avoid it from dominating the optimization owing to its sensitivity, is given lower weightage ($\lambda_{center} = 0.005$). The ablation results showing the isolated contribution of each loss function towards the accuracy of the model for the CIFAKE dataset is given in Table 7. These results show that no single loss function dominates others, and that their combination provides small yet complimentary gains.

Table 7. Ablation results for the contribution of loss functions (on the CIFAKE dataset)

Loss function	Accuracy (%)
Cross-entropy only	96.74
Cross entropy + Augmentation-Invariant loss	96.93
Cross entropy + Supervised Contrastive loss	96.99
Cross entropy+center loss	97.08
All loss functions	97.43

4.4.4. Ablation results for the per-augmentation performance of the model

In order to analyze the source of degradation in performance for each perturbation, an ablation study of each augmentation during testing has been performed on PRCG and one sample generator of GenImage (SDV1.4) datasets. The results are tabulated in table 8. The results show that the PRCG images suffer a significant drop in accuracy under noise augmentation, whereas the SDV1.4 dataset suffer during compression. This can be attributed to the fact that PRCG images being graphics rendered, are characterized by geometric artifacts like edges, etc which are disrupted by noise. On the other hand, SDV1.4 images are diffusion model generated. These are characterized by high frequency artifacts that are modified by compression.

Table 8. Per-augmentation performance of the model on CIFAKE dataset.

Augmentation	Accuracy for PRCG dataset (%) (std acc=96.25%)	Accuracy for GenImage dataset (%) (std acc =99.15%)
Compression	92.50% (-3.75%)	90.01% (-9.13%)
Blur	93.75% (-2.50%)	98.74% (-0.41%)
Color jitter	95.62% (-0.62%)	99.12% (-0.03%)
noise	80.00% (-16.25%)	95.11% (-4.04%)

5. Conclusion

The present work uses a multi-branch hybrid model to demarcate AI-generated images from natural images, by fusing two pretrained models – DINO and CLIP Vision Transformer models, with forensic and handcrafted feature modules via cross attention. These transformers capture the high-level semantic content and consistency, while the forensic and handcrafted modules extract low-level cues like texture, noise, and other pixel-level statistical cues. The model is further improved by optimizing multiple losses – augmentation-invariant loss, center loss and supervised contrastive loss. This helps in improving the discriminative cues marginally from all the branches of the model. Experimental results confirm that the model achieves high accuracy compared to baselines. Further, the model exhibits a nearly stable performance and robustness under the effect of different perturbations and a comparable improvement in cross-generator performance.

6. Limitations and Future Scope

Despite the performance improvement of the model compared to few baselines, the model has few limitations which show the scope for future work. While the proposed multi-branch model performs well on single datasets, it needs improvement on unseen generators as observed from fig.16. This indicates that the model still relies on generator-specific features. Future work can focus on developing domain-invariant features for cross-dataset generalization. In addition, the high parameter count, increased latency and memory consumption may limit the chances of its deployment in real time. Improvement in efficiency without sacrificing accuracy is needed in further development. Due to computational constraints of the proposed model, certain experiments had to be constrained to subsets of datasets. It may hence help to focus on improving the adversarial robustness of the model to random real-world image sources, and scaling the model to adapt to large and diverse datasets.

All the Declarations and Statements

Author Contribution Statement

Renuka Devi V S Bhamidipati – Conceptualization, Methodology, and Supervision: Proposed research ideas, and constructed the overall framework, Data Curation and Software Implementation: Handled data acquisition, dataset preprocessing, and implementing the research model. Model Training, Validation, and Performance Evaluation: Led the model training process, and validated results using standard metrics, Formal Analysis, Visualization, and Statistical Analysis: Performed in-depth analysis of experimental results and prepared performance charts. Writing – Drafted the initial manuscript, contributed to the literature survey, and documented the technical background of the study.
Srinivasa Rao Chanamallu – Conceptualization, Methodology, and Supervision: Proposed research ideas, constructed the overall framework, and supervised project execution. Model Training, Validation, and Performance Evaluation: benchmarked performance against existing methods, Formal Analysis, Visualization, and Statistical Analysis: ensured the statistical robustness of the evaluation, Writing – Review and Editing, and Project Management: Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.
Sudheer Gopinathan – Conceptualization, Methodology, and Supervision: Proposed research ideas, constructed the overall framework, and supervised project execution. Formal Analysis, Visualization, and Statistical Analysis: ensured the statistical robustness of the evaluation, Writing – Review and Editing, and Project Management: Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The authors declare no conflicts of interest.

Funding Declaration

This research received no specific grant from any funding agency.

Data Availability Statement

The datasets used in the proposed work are publicly available, and have been described in detail and appropriately cited in the manuscript. No new data was generated.

Ethical Declarations

This research does not involve any human subjects and/or animals.

Acknowledgements
We express our gratitude to the reviewers for their valuable recommendations, which have contributed to improving the quality of the paper.

Declaration of Generative AI in Scholarly Writing
No generative AI tools have been used in the preparation of the manuscript.

Abbreviations
The following abbreviations are used in this manuscript:

AI - Artificial Intelligence
DINO - Self-distillation with No Labels
CLIP - Contrastive Language-Image Pre-training
ViT - Vision Transformer
NI - Natural Image
CG - Computer Graphics
GAN - Generative Adversarial Network
CNN - Convolutional Neural Network
PIM - Photographic Image
LR - learning Rate
ROC - Receiver Operating Characteristic
AUC - Area Under the ROC Curve
TTA - Test Time Augmentation
MLP - Multi-Layer Perceptron
FFT - Fast Fourier Transform

Appendix
None.

References

[1] Brock A, Donahue J, Simonyan K. Large Scale Gan Training For High Fidelity Natural Image Synthesis. Int. Conf. Learn. Represent., 2019. https://doi.org/10.48550/arXiv.1809.11096.
[2] Lu Z, Huang D, Bai L, Qu J, Wu C, Liu X, et al. Seeing is not always believing: Benchmarking Human and Model Perception of AI-Generated Images. Proc. 37th Int. Conf. Neural Inf. Process. Syst., New Orleans, LA, USA: Curran Associates Inc.; 2023, p. 25435–47. https://doi.org/10.48550/arXiv.2304.13023.
[3] Gangu Rama Naidu, Chanamallu Srinivasa Rao. A CNN based Discrimination between Natural and Computer Generated Images. Pmj 2024;35:580–9. https://doi.org/10.52783/pmj.v35.i2s.2948.
[4] Nataraj L, Mohammed TM, Manjunath BS, Chandrasekaran S, Flenner A, Bappy JH, et al. Detecting GAN generated Fake Images using Co-occurrence Matrices. Electron Imaging 2019;31:532-1-532–7. https://doi.org/10.2352/ISSN.2470-1173.2019.5.MWSF-532.
[5] Baskar C, Govindasamy GP, Anbalagan S, Roomi SMM. Computer Graphic and Photographic Image Classification Using Transfer Learning Approach. Trait Signal 2022;39:1267–73. https://doi.org/10.18280/ts.390419.
[6] Castillo Camacho I, Wang K. Convolutional neural network initialization approaches for image manipulation detection. Digit Signal Process 2022;122:103376. https://doi.org/10.1016/j.dsp.2021.103376.
[7] Wang K. Self-Supervised Learning for the Distinction between Computer-Graphics Images and Natural Images. Appl Sci 2023;13:1887. https://doi.org/10.3390/app13031887.
[8] Ng T-T, Chang S-F, Hsu J, Xie L, Tsui M-P. Physics-motivated features for distinguishing photographic images and computer graphics. Proc. 13th Annu. ACM Int. Conf. Multimed., Hilton Singapore: ACM; 2005, p. 239–48. https://doi.org/10.1145/1101149.1101192.
[9] Zhang R-S, Quan W-Z, Fan L-B, Hu L-M, Yan D-M. Distinguishing Computer-Generated Images from Natural Images Using Channel and Pixel Correlation. J Comput Sci Technol 2020;35:592–602. https://doi.org/10.1007/s11390-020-0216-9.
[10] Gangan MP, K A, L LV. Distinguishing Natural and Computer-Generated Images using Multi-Colorspace fused EfficientNet. J Inf Secur Appl 2022;68:103261. https://doi.org/10.1016/j.jisa.2022.103261.
[11] Quan W, Wang K, Yan D-M, Zhang X, Pellerin D. Learn with diversity and from harder samples: Improving the generalization of CNN-Based detection of computer-generated images. Forensic Sci Int Digit Investig 2020;35:301023. https://doi.org/10.1016/j.fsidi.2020.301023.
[12] Grommelt P, Weiss L, Pfreundt F-J, Keuper J. Fake or JPEG? Revealing Common Biases in Generated Image Detection Datasets. In: Del Bue A, Canton C, Pont-Tuset J, Tommasi T, editors. Comput. Vis. – ECCV 2024 Workshop, vol. 15644, Cham: Springer Nature Switzerland; 2025, p. 80–95. https://doi.org/10.1007/978-3-031-92089-9_6.

[13] Guillaro F, Cozzolino D, Sud A, Dufour N, Verdoliva L. TruFor: Leveraging All-Round Clues for Trustworthy Image Forgery Detection and Localization. 2023 IEEECVF Conf. Comput. Vis. Pattern Recognit. CVPR, Vancouver, BC, Canada: IEEE; 2023, p. 20606–15. https://doi.org/10.1109/CVPR52729.2023.01974.
[14] Wang B, Wang F, Wang J, Yan H, Zhang F, Zhou S, et al. Unifaked: Universal Fake Image Detection Via Multiple Ensemble Learning 2025. https://doi.org/10.2139/ssrn.5139156.
[15] Jeong Y, Kim D, Ro Y, Choi J. FrePGAN: Robust Deepfake Detection Using Frequency-Level Perturbations. Proc AAAI Conf Artif Intell 2022;36:1060–8. https://doi.org/10.1609/aaai.v36i1.19990.
[16] Tan C, Zhao Y, Wei S, Gu G, Wei Y. Learning on Gradients: Generalized Artifacts Representation for GAN-Generated Images Detection. 2023 IEEECVF Conf. Comput. Vis. Pattern Recognit. CVPR, Vancouver, BC, Canada: IEEE; 2023, p. 12105–14. https://doi.org/10.1109/CVPR52729.2023.01165.
[17] Veselska O, Ziubina R. Reversible image steganography using transformer-based latent embedding. Adv Sci Technol Res J 2025;19:148–64. https://doi.org/10.12913/22998624/204419.
[18] Chen Y, Mai T, Wang S, Kang X. Computer-Generated Image Forensics Based on Vision Transformer with High-Frequency Feature Enhancement. 2025 33rd Eur. Signal Process. Conf. EUSIPCO, Palermo, Italy: IEEE; 2025, p. 1218–22. https://doi.org/10.23919/EUSIPCO63237.2025.11226696.
[19] Lamichhane D. Advanced Detection of AI-Generated Images Through Vision Transformers. IEEE Access 2025;13:3644–52. https://doi.org/10.1109/ACCESS.2024.3522759.
[20] Caron M, Touvron H, Misra I, Jegou H, Mairal J, Bojanowski P, et al. Emerging Properties in Self-Supervised Vision Transformers. 2021 IEEECVF Int. Conf. Comput. Vis. ICCV, Montreal, QC, Canada: IEEE; 2021, p. 9630–40. https://doi.org/10.1109/ICCV48922.2021.00951.
[21] Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, et al. Learning Transferable Visual Models From Natural Language Supervision. Proc. 38th Int. Conf. Mach. Learn., vol. 139, PMLR; 2021, p. 8748--8763. https://doi.org/10.48550/arXiv.2103.00020.
[22] Tan C, Zhao Y, Wei S, Gu G, Liu P, Wei Y. Frequency-Aware Deepfake Detection: Improving Generalizability through Frequency Space Domain Learning. Proc AAAI Conf Artif Intell 2024;38:5052–60. https://doi.org/10.1609/aaai.v38i5.28310.
[23] Alam I, Muneer MS, Woo SS. UGAD: Universal Generative AI Detector utilizing Frequency Fingerprints. Proc. 33rd ACM Int. Conf. Inf. Knowl. Manag., 2024, p. 4332–40. https://doi.org/10.1145/3627673.3680085.
[24] Cozzolino D, Poggi G, Verdoliva L. Recasting Residual-based Local Descriptors as Convolutional Neural Networks: an Application to Image Forgery Detection. Proc. 5th ACM Workshop Inf. Hiding Multimed. Secur., Philadelphia Pennsylvania USA: ACM; 2017, p. 159–64. https://doi.org/10.1145/3082031.3083247.
[25] Ming-Kuei Hu. Visual pattern recognition by moment invariants. IEEE Trans Inf Theory 1962;8:179–87. https://doi.org/10.1109/TIT.1962.1057692.
[26] Zhu M, Chen H, Yan Q, Huang X, Lin G, Li W, et al. GenImage: A Million-Scale Benchmark for Detecting AI-Generated Image. Proc. 37th Int. Conf. Neural Inf. Process. Syst., New Orleans, LA, USA: 2023, p. 77771–82. https://doi.org/10.48550/arXiv.2306.08571.
[27] Bird JJ, Lotfi A. CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. IEEE Access 2024;12:15642–50. https://doi.org/10.1109/ACCESS.2024.3356122.
[28] Ng T-T, Chang S-F, Hsu J, Pepeljugoski M. Columbia Photographic Images and Photorealistic Computer Graphics Dataset. ADVENT, Columbia University; 2005.
[29] Nichol A, Dhariwal P, Ramesh A, Shyam P, Mishkin P, McGrew B, et al. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. Int. Conf. Mach. Learn., arXiv; 2022. https://doi.org/10.48550/arXiv.2112.10741.
[30] Gu S, Chen D, Bao J, Wen F, Zhang B, Chen D, et al. Vector Quantized Diffusion Model for Text-to-Image Synthesis. 2022 IEEECVF Conf. Comput. Vis. Pattern Recognit. CVPR, New Orleans, LA, USA: IEEE; 2022, p. 10686–96. https://doi.org/10.1109/CVPR52688.2022.01043.
[31] Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B. High-Resolution Image Synthesis with Latent Diffusion Models. 2022 IEEECVF Conf. Comput. Vis. Pattern Recognit. CVPR, New Orleans, LA, USA: IEEE; 2022, p. 10674–85. https://doi.org/10.1109/CVPR52688.2022.01042.
[32] Dhariwal P, Nichol A. Diffusion Models Beat GANs on Image Synthesis. NIPS'21, vol. 672, Red Hook, NY, USA: Curran Associates Inc.; 2021, p. 8780–94. https://doi.org/10.48550/arXiv.2105.05233.
[33] Islam MT, Lee IH, Alzahrani AI, Muhammad K. MEXFIC: A meta ensemble eXplainable approach for AI-synthesized fake image classification. Alex Eng J 2025;116:351–63. https://doi.org/10.1016/j.aej.2024.12.031.
[34] Bartos GE, Akyol S. Deep Learning for Image Authentication: A Comparative Study on Real and AI-Generated Image Classification. 18th Int. Symp. Appl. Inform. Relat. Areas, Szekesfehervar, Hungary: n.d.
[35] Zhu M, Chen H, Huang M, Li W, Hu H, Hu J, et al. GenDet: Towards Good Generalizations for AI-Generated Image Detection. ArXiv 2023;abs/2312.08880.
[36] Chen J, Yao J, Niu L. A Single Simple Patch is All You Need for AI-generated Image Detection 2024. https://doi.org/10.48550/arXiv.2402.01123.

Authors' Profiles

Renuka Devi V. S. Bhamidipati is an Assistant Professor in department of Electronics and Communications Engineering at Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. She is currently pursuing her PhD at Jawaharlal Nehru Technological University of Kakinada. She has 19 years of teaching experience and her areas of interest include Image Processing, Digital Forensics and Machine Learning.

Srinivasa Rao Chanamallu is a professor of Electronics and Communications Engineering and Principal at Jawaharlal Nehru Technological University of Kakinada, Narsaraopet Campus. His qualifications are as mentioned. Ph.D, M.Tech. and B.Tech. He has 31 years of teaching experience and his areas of interest include Image and Video forensics, machine learning and Evolutionary computing. His total number of research publications is 81.

Sudheer Gopinathan received his PhD in Applied Mathematics from Andhra University in 2006. He is currently a professor in the Department of Mathematics at Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam. His current research interests include Image Processing, Ultrasonic NDT, pseudo-spectral methods and Wavelet Analysis.

How to cite this paper:

Venkata Satya Renuka Devi Bhamidipati, Srinivasa Rao Chanamallu, Sudheer Gopinathan, "A Multi-Branch Transformer Based Cross-Attention Framework for Computer-Generated Image Detection", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.18, No.4, pp. 25-47, 2026. DOI:10.5815/ijigsp.2026.04.02