Journal of Engineering and Technology for Industrial Applications

ISSN ONLINE: 2447-0228

ITEGAM-JETIA

Manaus, v.12 n.58, p. 304-314. March/April, 2026.
DOI: https://doi.org/10.5935/jetia. v12i58.3045

RESEARCH OPEN ACCESS

ENHANCED DEEPFAKE IMAGE DETECTION VIA SWIN TRANSFORMER WITH VISUAL ATTENTION ANALYSIS

Vineela Krishna Suri*[1] and Prasad GVSNRV[2]

[1]Research Scholar, Department of CSE, JNTUK, Kakinada, Andhra Pradesh, India
[2]Professor & Director(PGCRD), Department of CSE, SRGEC, Gudlavalleru, Andhra Pradesh, India.

[1]http://orcid.org/0000-0002-4563-1979, [2]http://orcid.org/0000-0003-0679-7324

Email: *vineela.suri@gmail.com, gutta.prasad1@gmail.com

ARTICLE INFO

Article History
Received: December 2, 2025
Reviewed: January 6, 2026
Accepted: January 13, 2026
Published: March 31, 2026

Keywords:
Deepfakes,
Vision Transformers,
Convolutional Neural Networks.

ABSTRACT

Deepfakes, synthetic media created using advanced machine learning techniques, pose significant societal challenges by spreading misinformation and undermining trust in media. With the increasing sophistication of deepfake technologies, distinguishing between genuine and synthetic media has become increasingly difficult. This paper presents a robust deepfake image detection framework using the Swin-B Transformer, a pre-trained model fine-tuned for our application. By integrating a hybrid dataset that combines real images from the FFHQ dataset and synthetically generated fake images from a publicly available Kaggle dataset, we simulate real-world media scenarios. Our model achieves an impressive accuracy of 97.47% on the test set, demonstrating superior generalization to both real and synthetic visual data. Using Grad-CAM, we visualize the spatial segments of the image that the model focuses on during classification, providing insight into the decision-making process. This work contributes to enhancing content authenticity, controlling fake news, and ensuring digital trust and safety.

I. INTRODUCTION

The pioneering work of Christoph Bregler, Michele Covell, and Malcolm Slaney in developing the video rewrite program in 1997 laid the foundation for Deepfakes. It is marked as the first automated approach to facial reanimation, leading to the more sophisticated deepfake technologies that we encounter today. The term deepfake has since come to encompass both these fake media and the specific method of their creation. The manipulation of digital images and videos has been a promising technique for many years and is done through the use of visual effects. However, the inception of deep learning architectures like Generative Adversarial Networks (GANs) has led to a marked increase in the realistic nature of fake content and the ease with which they can be generated [1], [2]. These AI-generated media, commonly referred to as deepfakes, are easily produced using artificial intelligence tools. Through the use of deep learning algorithms, it is now possible to generate fabricated videos and images that are virtually challenging to differentiate from real ones. Since their emergence in 2017, there has been a proliferation of open source deepfake generation methods and tools, resulting in a growing number of synthetic media clips.

FakeApp [3] and FaceSwap [4] are two prominent publicly accessible deepfake implementations. Throughout history, there have been numerous instances in which deepfakes have been utilized for malicious purposes, such as creating political tensions, staging fake terrorism events, producing pornographic content, and blackmailing.

As an example, University of Maryland researchers created a bot by the name of Katie Jones on the LinkedIn platform. In order to add more authenticity to this fake profile, a profile picture generated using deepfake technologies was also added, and the bot was then left to make connections on the platform. In just two months, this profile was connected with two US state senators and one Department of Defense (DoD) personnel [5]. It is estimated that, bots make up 65% of the Internet currently. With the advent of deepfake technologies, it might become even more difficult to detect fake identities on the Internet. A study by the AI firm Deeptrace revealed that only in the month of September 2019 they identified 15,000 deepfake videos online, nearly double the increase over the previous nine months.

Of these videos, 96% were pornographic in nature, among which 99% of those targeted female celebrities, overlaying their faces on the bodies of some porn stars [6]. This marked the significance of robust and reliable methods for detecting deepfakes thus preventing their abuse. At the heart of deepfakes lies the intricate process of facial swapping, a technique that allows for seamless overlaying of facial images of a target individual onto videos of a source individual as shown in Figure1 This mesmerizing artistry results in videos of the target individual appearing to say or do things that were originally performed by the source individual as shown in Figure 2. An example of the influence of deepfakes can be seen in the circulation of such videos related to the Russian-Ukraine conflict, including a deepfake of Russian President Vladimir Putin declaring peace.

Figure 1: Original Image and Corresponding Deepfake Image.
Source: Authors, (2026).

Figure 2: An example Deepfake video.
Source: Authors, (2026).

The rapid growth of generative models has led to the extensive creation and spread of deepfake images. Although these technologies have creative and commercial applications, they also carry serious risks such as the spread of false information, identity theft, and a decline in public confidence in digital content. A crucial challenge in the areas of computer vision and social computing is the detection of these misleading media content. The increasing societal impact of deepfake images is another factor driving this study. As deepfakes significantly influence public opinion, identity verification and social media integrity, strong detection techniques contribute directly to the broader goals of social computing there by improving digital trust and authenticity of the content.

I.1 OBJECTIVES

The main objectives of our proposed work are the following.

1. Create a hybrid dataset that combines real images from the FFHQ dataset with synthetically generated fake images from a publicly available Kaggle dataset, representing diverse and challenging real-word scenarios.
2. Use, optimize, and evaluate the Swin-B transformer for deepfake image detection, a variant not extensively applied in this context.
3. Show that our approach has better generalization to heterogeneous data and outperforms several other existing approaches.

II. LITERATURE SURVEY

In the earliest methods, engineered features obtained from the inconsistencies and artifacts of fake images were used for classification. In the paper [7] make use of an image pre-processing step, where they apply Gaussian blur and Gaussian noise to the images in order to extract low-level high frequency cues of GAN generated images. This improves pixel-level statistical similarity between authentic and fraudulent images, allowing forensic classifiers to learn more intrinsic and valuable characteristics compared to existing techniques or networks.

In the paper [8], identified several distinguishing features such as missing reflection in the eye, loss of geometry and shape, difference in colour of two eyes of the same person etc., and then created a detection pipeline which segments all the various important features like iris, face crops etc. These segmented and extracted features were tested for authenticity and used for classification. They achieved their best accuracy of 83.8% with eyes as the primary mode of classification.

Hsu et al. introduced a two-phase deep learning method for the detection of deepfakes [9]. It makes use of the Siamese Network architecture based on the Common Fake Feature Network (CFFN) for feature extraction. The Siamese network takes pairs of images along with a pairwise label during training. When both images within a pair are genuine, the label is designated as 1, indicating a ”real-real” pairing. Conversely, if the images belong to different classes, meaning that one is authentic and the other one is manipulated, the label is assigned as 0, reflecting a ”fake-real” or ”real-fake” pairing.

During the training phase, a pairwise label is provided, and the model learns the distinguishing features of each class. Further, it can also understand the representative characteristics of real and fake images. Once the training is completed, a base image, whose class is known (real or fake), is passed along with the input image given by the user. Depending on the output obtained and the type of base image, we can determine the class of input image. The major setback of this approach is that it uses a non-standard dataset which is generated only using GANs. Hence, cannot work well with other deepfake implementations and can provide results only for GAN generated images.

Another method using pairwise learning was proposed by Zhao et al. for deepfake detection [10], where they made use of the self-consistency of local source features noticed in deepfakes, which are content-independent and spatially local information of images. These local features can be the result of imaging pipelines or encoding methods used in different image synthesis approaches. A CNN-based model is used to learn pairwise self-consistency. The hypothesis behind this paper is that the original image will have consistent features in different localities and a modified image will be inconsistent in generated and original portions of the image. The model is trained using representation learning technique, where, the model, instead of learning about the overall features, looks only at parts of the images.

This approach differentiates between genuine and manipulated images by penalizing feature vector pairs with low similarity scores within the same image. Conversely, it also penalizes the pairs of feature vectors from different images for exhibiting a higher similarity score. In this paper, cosine similarity is used as an evaluation metric. The feature maps being learned were then provided as input to a classification method for deepfake detection. Although this model performs well on locally modified images, it struggles with fully generated fake images, where features remain consistent at all positions within the image. Zhou et al. propose a different approach [11], in which they make use of a two-stream neural network. Instead of pairwise learning to understand latent relations between pairs of images, one stream is used to detect faces and determine their genuineness.

The other stream is used to detect traces which might be left due to any steganographic tampering. They make use of FaceSwap and SwapMe apps available on the App Store in iOS to create a custom dataset. However, the customized dataset used in their research is out-dated and the results are not updated with recent GAN-based generation methods. Moreover, the evaluation was primarily theoretical and when tested against a standard Celeb-DF dataset, achieved an AUC score of only 55.7%. A simpler approach was introduced by [12], where they implemented a modified VGG-16 network called as NA-VGG (N: Noise, A: Augmentation). Here, the input is first processed with an SRM filter which highlights image noise.

SRM stands for Style Recalibration Module, which is a fixed weight matrix acquired by training a shallow neural network to modify the image to a specific type, for example, from RGB to HSV etc. In this paper, the SRM layer extracts and highlights latent artifacts in the image which go unnoticed in the case of RGB. After generating this version of the input, augmentation is performed, where more images were generated by flipping, rotating, adjusting the brightness etc. of the image. The augmented images are passed to a standard VGG-16 CNN which contains 16 layers in total: 13 convolution layers and 3 fully connected layers. In order to downsize the resolution, 5 pooling layers are also used intermediately after convolutions.

In their research, they used the Celeb-DF dataset which is created by swapping faces in the CelebA-HQ dataset. Theerthagiri introduced an approach using InceptionNet [13], a CNN based architecture to capture multiscale features to identify deepfakes. In their study, they used a public dataset downloaded from kaggle consisting of 401 videos. Data Augmentation is applied to generate 3745 images from the dataset. The efficacy of the model was evaluated using various metrics derived from the confusion matrix, achieving an accuracy of 93% in recognizing deepfake images and videos. Another method was introduced by pasupuleti [14], using a Custom Densenet architecture, which is a deep residual neural network.

To classify images as *real* or *fake* they employed a CNN based architecture trained using a binary cross-entropy function. The performance of the model is evaluated using various metrics calculated from the confusion matrix. Ghita et al. proposed an approach for detecting deepfakes using Vision Transformers(ViT) [15]. This ViT based detection model is trained with a dataset consisting of 40000 samples including both real and deepfake images collected from the Kaggle dataset. The model achieved a score of 0.899 on a dataset consisting of 40000 samples. Their initial experiments emphasized the need for a large dataset for training and the fast convergence of the model. Compared to other baseline deepfake detection techniques, the performance of the ViT model was consistent with existing research, indicating its potential for further investigation.

Authors of [16] suggested a shallow vision transformer for recognizing deepfakes, incorporating both an attention mechanism and multi-head attention module. The attention mechanism highlights the critical parts of images, while the multi-head attention module determines how much attention should be allocated to local-level features. Finally, the softmax layer classifies an image as *real* or *not real*. Experiments conducted on two datasets-Real Fake Face (RFF) and Real and Fake Face Detection (RFFD) datasets show that the model achieves higher accuracy of 0.92 on RFF and 0.89 on RFFD datasets. The results outperforms most of the existing models such as GoogleNet, XceptionNet, ResNet50 and other baseline vision transformers. Furthermore, the model demonstrated an accuracy of 0.90 even when trained only on 50% of the RFF dataset.

II.1 LIMITATIONS OF EXISTING METHODS

- Most of the existing systems utilize standard versions of visual transformers or CNN architectures, like EfficientNet, ResNet variants, which may not capture global contextual clues present in high-quality deepfake images.
- They often utilize a single-source benchmark dataset or domain-specific image collections, which may not fully reflect the diversity and complexity of real-world content shared on social platforms.
- Although Swin Transformers have been used in some earlier studies, the Swin-B -which is a larger and more expressive variant of the Swin Transformer, has not been investigated for deepfake image detection.

These days Deepfakes have become very realistic, making local artifacts difficult to detect. In this context, there is a need for a transformer-based model which will capture global dependencies and can model long-range dependencies of images [16].

The self-attention mechanism in the transformer architecture helps focus attention on minute anomalies that CNNs may overlook. When fine-tuned, they offer better generalization and are robust to the new manipulation methods available. To mitigate these issues, we propose a novel deepfake detection model that leverages the Swin-B Transformer, a high-capacity hierarchical vision transformer, applied to a combined dataset consisting of real images from the FFHQ dataset and fake images sourced from a publicly available Kaggle dataset. This hybrid dataset simulates real-world media environment, improving the model's ability to generalize to unseen manipulations. Our method is novel in two ways: (i) we created a mixed dataset with both real and fake images from two different sources to represent realistic media scenarios; and (ii) we applied and assessed the performance of the Swin-B Transformer, an architecture that hasn't been studied in this context before.

III. DATASET DESCRIPTION

We constructed a hybrid dataset, consisting of images collected from two different datasets. Real images were collected from FFHQ Dataset [17] and DeepFake images from a dataset available on Kaggle [18]. Flickr-Faces-HQ (FFHQ) is an image dataset consisting of 70,000 high-quality PNG images of real human faces each with a resolution of 1024 × 1024, originally created as a standard for GANs [17]. The images in the data set were originally extracted from Flickr.com, an online platform for hosting images and videos, and cropped using the Dlib library [19]. The second dataset [18], is an image dataset consisting of 95,000 JPG images of forged human faces each with a resolution of 256 × 256. Images extracted from both datasets were combined and organized into two main directories: real and deepfake as shown in Figure 3. This combined dataset simulates the variability seen in real-world social media content. Samples of the real and deepfake images from the dataset are shown in Figure 4a and Figure 4b.

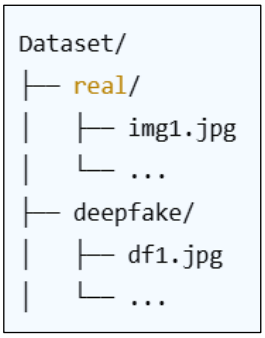

Figure 3: Structure of the dataset.
Source: Authors, (2026).

(a) (b)

Figure 4: Example Real (a) and Deepfake, (b) images from the dataset.
Source: Authors, (2026).

IV. PROPOSED MODEL

Transformers have shown immense capabilities and work especially well in tasks where pixel-level details might matter. Our proposed model uses the Swin - B (Base) Transformer, a mid-range model which is a specific type of Swin Transformer designed to provide a good balance of performance, computational cost and speed. Table 1 compares various Swin Transformer variants with respect to their computational costs. Among all the variants of Swin, Swin-B was chosen because of its superior ability to balance hierarchical representation learning, global context modeling, and computational efficiency [20]. Swin-B uses shifted window self-attention to learn local and non-local visual patterns with a multi-scale feature hierarchy unlike traditional CNNs, which use fixed receptive fields and have difficulties in capturing long-range dependencies.

Swin-B is much less computationally expensive than regular Vision Transformers, which exhibit quadratic computation complexity as a function of image resolution, making it effectively able to process high-resolution images, available in social media content, with smaller computational expense. Pre-trained weights were used before training so that the model requires less training times. In addition, it also helps in avoiding overfitting, as it can be trained for fewer epochs. After training, we evaluated the efficiency of the model by calculating important metrics on the test set. The model is then deployed for demonstration using the Gradio API, where it can be tested by giving an input image (real/fake).

Table 1: Swin Transformer variants with Computational costs.

Model	Params (M)	Computational Cost (FLOPs, G @224×224)	Depths	Embed Dim	Heads (Max)	Window Size	ImageNet Top-1 (%)
Swin-T	28	4.5	[2, 2, 6, 2]	96	24	7×7	81.3
Swin-S	50	8.7	[2, 2, 18, 2]	96	24	7×7	83.0
Swin-B	**88**	**15.4**	**[2, 2, 18, 2]**	**128**	**32**	**7×7**	**83.5**
Swin-L	197	34.5	[2, 2, 18, 2]	192	48	7×7	84.8

Source: Authors, (2026).

The overall architecture of the model is broken down into 5 modules shown in Figure 5. They are:

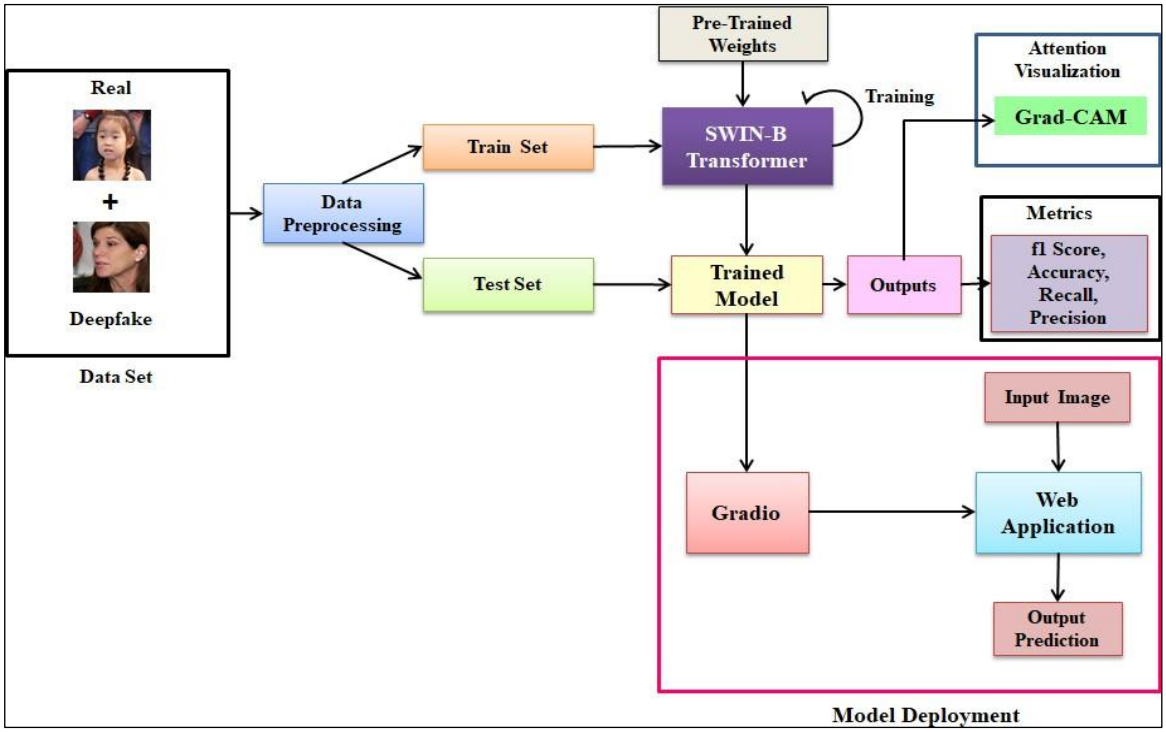

Figure 5: Proposed System architecture.
Source: Authors, (2026).

IV.1 DATA PRE-PROCESSING

All the images present in the hybrid dataset are transformed so that they are in a fixed size of 224×224 pixels - the expected input image size for the model, and are normalized. In order to balance both the classes present in the dataset, and to avoid overfitting, data augmentation is applied on real face images. The dataset was divided into subsets, with 70% used for training and 30% used for testing.

IV.2 MODEL TRAINING

After splitting the extracted image dataset into training and testing subsets, we start the training phase. Here, we make use of the pre-trained Swin-B Transformer model. Initially, load the Transformer pre-trained on ImageNet. The pre-trained weights for this model are obtained from the official weights submitted by Microsoft to the Huggingface platform [21]. This model contains roughly 28 million trainable parameters and is pre-trained on more than 14 million images present in ImageNet dataset. The Swin-B Transformer model contains four blocks called "swin layers", each performing similar operations. In the first layer, the image is broken down into non-overlapping patches of size 4 x 4 pixels resulting in 56 x 56 tokens. Swin divides the 56 x 56 tokens into small windows of size 7 x 7.

Unlike conventional Vision Transformers, where multi-head attention is calculated globally for the entire image taking a lot of processing time, Swin transformers applies attention only inside local attention window of size 7 x 7. This approach is known as Window-based Multi-head Self Attention (W-MSA) as shown in Eq(1). In the next layer, the windows were shifted by patch size of 3 supporting cross-window communication called as Shifted Window Multi-head Self Attention (SW-MSA). After each group of Swin blocks, a patch merging operation is performed, where 2×2 neighbouring patches are concatenated and linearly transformed there by increasing feature depth while decreasing spatial resolution. The overall architecture of the Swin-B transformer consisting of four stages, is illustrated in Figure 6 and the functionality of each stage is summarized in Table 2.

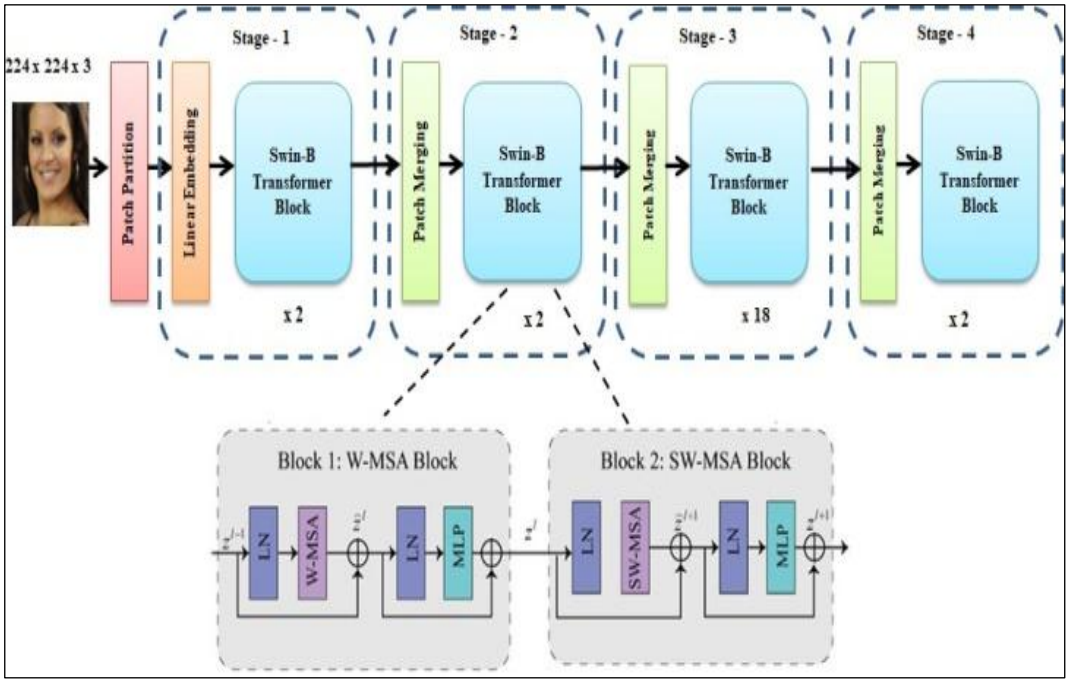

Figure 6: Architecture of Swin-B Transformer.
Source: Authors, (2026).

Table 2: Layers inside each Stage of Swin-B Transformer.

Stage	Input Size	Layers	Hidden Dim	Operations
1	56×56 (after 4×4 patch split)	2	128	W-MSA + SW-MSA
2	28×28 (after Patch Merging)	2	256	W-MSA + SW-MSA
3	14×14	18	512	W-MSA + SW-MSA
4	7×7	2	1024	W-MSA + SW-MSA

Source: Authors, (2026).

$$MultiHead(Q,K,V) = [head_1, \dots, head_h]W_0 \quad (1)$$

Where, headi is self-attention, calculated using Eq.(2) and Eq.(3)

$$head_i = Attention\left(Q\,W_i^Q, K\,W_i^K, V\,W_i^V\right) \quad (2)$$

$$Attention(Q,K,V) = Softmax\left(\frac{(QK^T)}{\sqrt{d_k}}\right)V \quad (3)$$

Where Queries $Q \in R^{n\times dk}$,Keys $K \in R^{n\times dk}$,Values $V \in R^{n\times dv}$. The final output layer of the Swin-B backbone produces a feature vector f $\in R^d$ and the fully connected layer converts f to class scores as shown in Eq(4).

$$= Wf + b \quad (4)$$

Where $W \in R^{n\times d\infty}$, $b \in R^n$ is the number of classes. Softmax is then applied to z which classifies the input into categories: Real, DeepFake. The softmax function is given by Eq(5):

$$\hat{y}_i = \frac{e^{z_i}}{\Sigma_{j=1}^{n} e^{z_j}} \quad (5)$$

Where $\hat{y}_i$ is the predicted probability for class i. Next, we calculate the loss using Cross Entropy function as shown in Eq(6). We take the cross-entropy loss and perform back propagation using gradient descent to modify the weights of the model.

$$Cross - Entropy = -\sum_x p(x)\log(q(x)) \quad (6)$$

Where p(x) is the true distribution, q(x) is the probability being predicted. In this way, the model is trained for 10 epochs. The total cross entropy for each epoch is observed as shown in Table 3.

Table 3: Cross-entropy loss for training and test set after each epoch.

Epoch Number	Training Loss	Testing Loss
1	0.4452	0.1691
2	0.1909	0.1585
3	0.1380	0.1465
4	0.1156	0.1363
5	0.0945	0.1338
6	0.0802	0.1298
7	0.0713	0.1241
8	0.0663	0.1231
9	0.0403	0.1212
10	0.0242	0.1210

Source: Authors, (2026).

IV.3 ATTENTION VISUALIZATION

Using Gradient-weighted Class Activation Mapping (Grad-CAM) technique, as proposed in [22], heatmaps are generated to visualize the spatial segments of the image on which the proposed model focuses during the classification of real and deepfake images by fusing the feature activations and the corresponding gradients from the layers being selected. These heatmaps were overlaid using a JET color map featuring red, yellow, blue, and green colors, where each color represents the degree of importance of the features:

- Red/Yellow: High Activation, most important regions that strongly influence the models decision.
- Green: Moderate Activation, regions that contribute partially to the classification.
- Blue: Low Activation, regions that have minimal or no influence on the classification.

Figure 7 and Figure 8 illustrate the Grad-CAM heatmaps generated for real and DeepFake images at all stages. The real image heatmap shows smooth, consistent attention on natural facial regions, and the activation is consistent from stage-1 onward. The fake image heatmap shows more scattered, irregular, and intense activations around unnatural boundaries and textures, with later stages (stage 2 and stage-3) highlighting these artifacts even more.

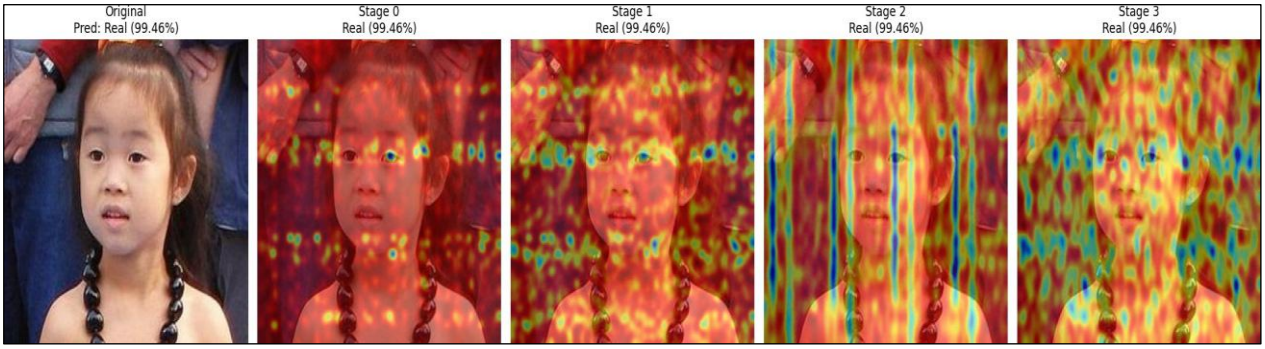

Figure 7: Grad-CAM heatmap generated from Stage 0 to Stage 3 for real image.
Source: Authors, (2026).

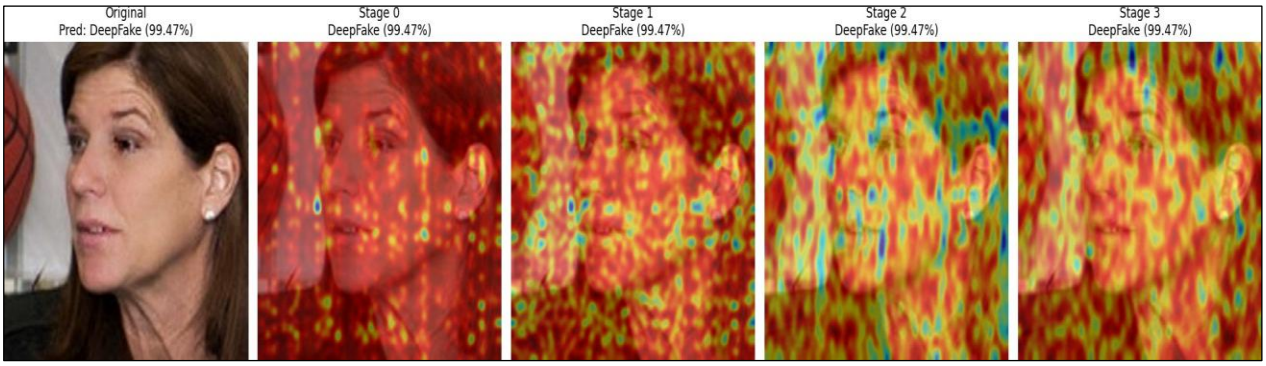

Figure 8: Grad-CAM heatmap generated from Stage 0 to Stage 3 for deepfake image.
Source: Authors, (2026).

IV.4 TESTING AND EVALUATION

Once the model has been sufficiently trained, its performance is then evaluated on the test set using standard evaluation metrics.

IV.5 DEPLOYMENT

Finally, after testing and evaluation, the model is deployed using Gradio API [23]. We created a Gradio Interface object that accepts a function with two parameters: inputs to the function and output from the function. You can quickly deploy a demo application by calling the launch() method on the interface object. The input and output objects can be of various types like text, images, audio, videos, buttons, numbers, etc. In our application, the input is a PIL image, and the output is the predicted class label. The prediction function contains the logic for generating predictions from an input image. When a user uploads an image, the API invokes this function, where the image is first processed using an AutoImageProcessor to ensure that the image is resized to 224 x 224 pixels - the expected input image size for the model. The processed image is then passed to the model to find the predictions, which are then mapped to their corresponding classes. The predictions are formatted appropriately before being returned to the API and displayed to the user. Figure 9 shows the User Interface where a random image from the "real" class of the test set was passed as an input. Figure 10 shows the output of a fake image from the dataset, which is correctly classified as Deepfake.

Figure 9: Output of a real image.
Source: Authors, (2026).

Figure 10: Output of a fake image.
Source: Authors, (2026).

V. EVALUATION OF RESULTS

The evaluation is performed on the test set after each epoch. The test set consists of 93,590 images in total. The model's performance is evaluated using four metrics. They are: Accuracy, Precision, Recall and F1 score. To carry out this work, we first extracted images from dataset composed of real faces from FFHQ real faces dataset [17] and fake images from kaggle deepfake images dataset [19]. These images are used to train our model on both real and deepfake images. We initialized our model with pre-trained weights to reduce training time and to improve the model's performance. We ran all of our tests on Google Colab Pro, an open-source Python notebook environment featuring a Tesla T4 GPU with 16GB of VRAM and approximately 13GB of system RAM.

To ease the T4 GPU's memory constraints, we used mixed precision training using 'torch.cuda.amp', which helped us in reducing both training time and VRAM usage. We saved checkpoints after every epoch to resume training if Colab disconnects. Furthermore, the model is optimized using Adam optimizer with a learning rate of 0.0001, $\beta 1 = 0.9$, and $\beta 2 = 0.999$ to ensure more precise parameter tuning. After training for 10 epochs, we evaluated performance using various metrics as shown in Figure 11. A comparison with the existing systems, shown in Table 4, indicates that our model performs well in most cases.

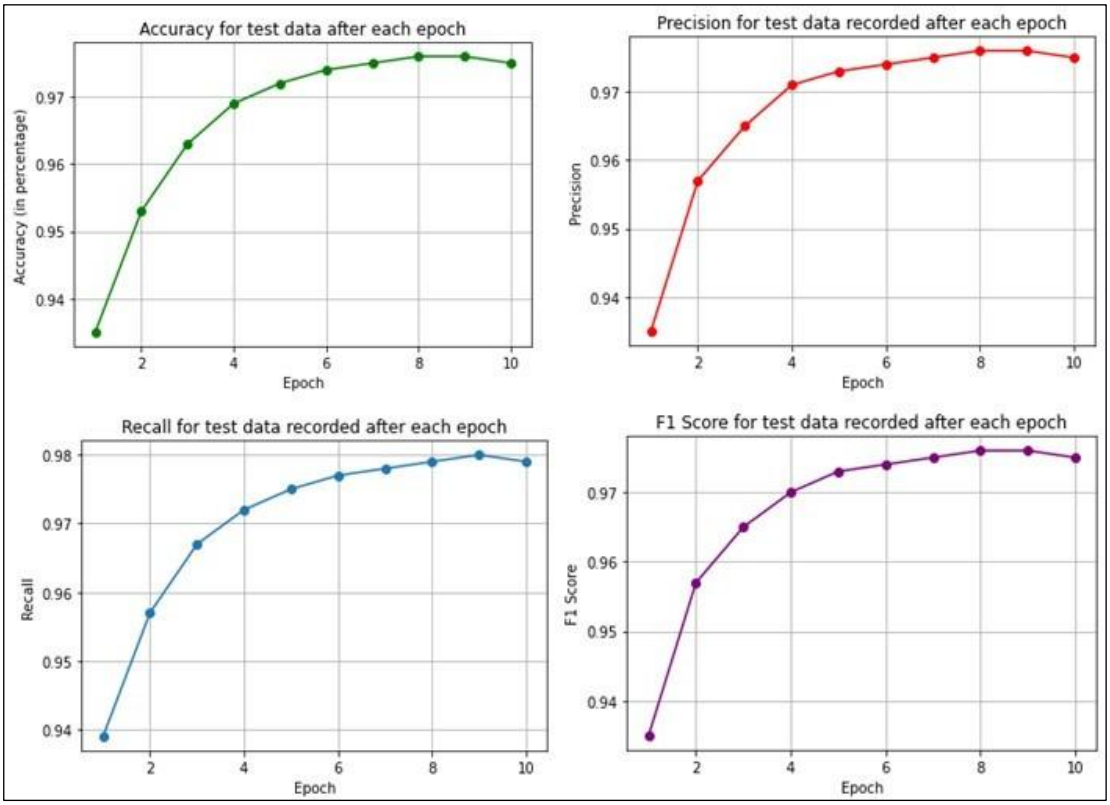

Figure 11: A graph showing the Accuracy, Precision, Recall and F1-Score on test data after each epoch.
Source: Authors, (2026).

Table 4: Comparison between Existing System and Our Proposed System.

Method	**Accuracy**	**Precision**	**Recall**	**F1-score**
Two Stream Neural Networks by Zhou et al.	55.7%	-	-	-
Deepfake Detection by Hsu et al.	-	90.9%	86.5%	-
VGG Network by Chang et al.	85.7%	-	-	-
InceptionNet by Theerthagiri	93%	-	-	-
Custom Densenet by Pasupuleti	97%	95%	-	75%
Vision Transformer by Ghita	89.9%	-	-	-
Proposed System	**97.47%**	**97.4%**	**97.56%**	**97.5%**

Source: Authors, (2026).

VI. ETHICAL AND SOCIAL IMPACT

As synthetic media technologies evolve, deepfake images have become a major societal and ethical challenge, especially on social computing platforms where images are frequently shared, altered, and used for other purposes. In most of the social computing platforms information spreads rapidly and user-generated content is often trusted without verification. The ability to generate realistic but deepfake human faces increases the concerns about misinformation, identity fraud, defamation, influence public opinion leading to the undermine public trust in visual media.

Our research technically addresses these challenges by employing Swin-B Transformer on a hybrid dataset combining real images from FFHQ and fake images from a publicly available Kaggle dataset. The dataset used in our approach reflects the diverse and real-world nature of social media content, which improves generalization and increases detection accuracy in real-world scenarios. This work has several important societal benefits, including:

- **Enhanced Content Authenticity:** Assisting social platforms to detect and filter deepfake images before it misleads users.
- **Support for the control of Fake News:** By detecting deepfake images early, can limit the viral spread of this manipulated visuals particularly in political or sensitive contexts.
- **Ensuring Digital Trust and Safety:** Allows Governments, institutions, platforms and users to verify the legitimacy of images there by improving public confidence in online content.

VII. CONCLUSION

In this paper, we addressed the growing issue of deepfake image generation and its potential negative societal impacts. Deepfakes are an increasing concern due to their ability to spread misinformation, support harassment, and undermine public trust in the visual media. To combat this, we proposed a robust detection framework using Swin-B, a standard variant of the Swin Transformer, which has demonstrated optimal performance in computer vision tasks compared to traditional CNNs. By leveraging the capabilities of this pre-trained model and fine-tuning it for deepfake detection, we achieved an accuracy of 97.47% on the test set of our hybrid dataset.

Furthermore, we used Grad-CAM to analyze which segments of the image the model is focusing on while classifying the images as *Real* or *DeepFake*. By promoting safer and more trustworthy digital environments, this work not only improves technical accuracy in the detection of deepfakes, but also supports the objectives of social computing.As the deepfake generation techniques continue to evolve, it will be necessary to re-access the model's performance and potentially retrain our model to ensure its effectiveness. Future work will focus on integrating temporal or multimodal features, extending the detection to deepfake videos, and continuously updating our model with the newly synthetic images sustaining its accuracy and reliability.

Availability of data and materials

"The data that support the findings of this study can be downloaded from https://github.com/NVlabs/ ffhq-datasetand; https://www.kaggle.com/datasets/manjilkarki/deepfake-and-real-images"

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

Funding

This research received no external funding.

VIII. AUTHOR'S CONTRIBUTION

Conceptualization: Vineela Krishna. Suri, Prasad. GVSNRV.
Methodology: Vineela Krishna. Suri, Prasad. GVSNRV.
Investigation: Vineela Krishna. Suri, Prasad. GVSNRV.
Discussion of results: Vineela Krishna. Suri, Prasad. GVSNRV.
Writing – Original Draft: Vineela Krishna. Suri, Prasad. GVSNRV.
Writing – Review and Editing: Vineela Krishna. Suri, Prasad. GVSNRV.
Resources Vineela Krishna. Suri, Prasad. GVSNRV.
Supervision: Vineela Krishna. Suri, Prasad. GVSNRV.
Approval of the final text: Vineela Krishna. Suri, Prasad. GVSNRV.

IX. REFERENCES

[1] M. -Y. Liu, X. Huang, J. Yu, T. -C. Wang and A. Mallya, "Generative Adversarial Networks for Image and Video Synthesis: Algorithms and Applications," in Proceedings of the IEEE, vol. 109, no. 5, pp. 839-862, May 2021, doi: 10.1109/JPROC.2021.3049196.

[2] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. 2022. Transformers in Vision: A Survey. ACM Comput. Surv. 54, 10s, Article 200 (January 2022), 41 pages. https://doi.org/10.1145/3505244

[3] FakeApp, "FakeApp: Deepfake Creation Tool," [Online]. Available: https://www.fakeapp.org/.FaceSwap, "Deepfakes FaceSwap GitHub Repository," [Online]. Available: https://github.com/ deepfakes/faceswap.

[4] FaceSwap, "Deepfakes FaceSwap GitHub Repository," [Online]. Available: https://github.com/ deepfakes/faceswap.

[5] J. Vincent, "AI-generated fake faces are being used to trick people online," *The Verge*, Jun. 13, 2019. [Online]. Available: https://www.theverge.com/2019/6/13/18677341/ ai-generated-fake-faces-spy-linked-in-contacts-associated-press

[6] A. Hern, "What are deepfakes – and how can you spot them?" *The Guardian*, Jan. 13, 2020. [Online]. Available: https://www.theguardian.com/technology/2020/jan/13/ what-are-deepfakes-and-how-can-you-spot-them.

[7] X. a. P. B. a. W. W. a. D. J. Xuan, "On the Generalization of GAN Image Forensics," in Biometric Recognition: 14th Chinese Conference, CCBR 2019, Zhuzhou, 2019.

[8] F. Matern, C. Riess and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," 2019 IEEE Wintere Applications of Computer Vision Workshops (WACVW), pp. 83-92, 2019

[9] C.-C. Hsu, Y.-X. Zhuang and C.-Y. Lee, "Deep Fake Image Detection Based on Pairwise Learning," Applied Sciences, vol. 10, p. 370, 2020.

[10] T. a. X. X. a. X. M. a. D. H. a. X. Y. a. X. W. Zhao, "Learning Self-consistency for Deepfake Detection," in IEEE/CVF Conference on Computer Vision (ICCV), 2021.

[11] P. Zhou, X. Han, V. I. Morariu and L. S. Davis, "Two-stream Neural Networks for Tampered Face Detection," in 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017.

[12] X. Chang, J. Wu, T. Yang and G. Feng, "DeepFake Face Image Detection based on Improved VGG Convolutional Neural Network," 39th Chinese Control Conference, pp. 7252-7256, 2020.

[13] P. Theerthagiri and G. b. Nagaladinne, ”Deepfake Face Detection Using Deep InceptionNet Learning Algorithm,” 2023 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS), Bhopal, India, 2023.

[14] V. R. Pasupuleti, P. Reddy Tathireddy, G. Dontagani and S. A. Rahim, ”Deepfake Detection Using Custom Densenet,” 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-5, doi: 10.1109/ICCCNT56998.2023.10307521.

[15] B. Ghita, I. Kuzminykh, A. Usama, T. Bakhshi and J. Marchang, ”Deepfake Image Detection Using Vision Transformer Models,” 2024 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Tbilisi, Georgia, 2024, pp. 332-335, doi: 10.1109/BlackSeaCom61746.2024.10646310.

[16] Usmani, S., Kumar, S. & Sadhya, D. Efficient deepfake detection using shallow vision transformer. Multimed Tools Appl 83, 12339–12362 (2024). https://doi.org/10.1007/s11042-023-15910-z

[17] NVLabs, ”Flickr-Faces-HQ (FFHQ) Dataset,” [Online]. Available: https://github.com/ NVlabs/ffhq-dataset

[18] M. Karki, ”Deepfake and Real Images,” Kaggle, [Online]. Available: https://www.kaggle.com/ datasets/manjilkarki/deepfake-and-real-images.

[19] Karras, Tero, et al. ”A Style-Based Generator Architecture for Generative Adversarial Networks.” CVPR 2019.

[20] Liu, Ze & Lin, Yutong & Cao, Yue & Hu, Han & Wei, Yixuan & Zhang, Zheng & Lin, Stephen & Guo, Baining. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. 9992-10002. 10.1109/ICCV48922.2021.00986.

[21] Hugging Face, ”Swin Transformer,” [Online]. Available: https://huggingface.co/docs/ transformers/model_doc/swin

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, ”Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization,” 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 618-626, doi: 10.1109/ICCV.2017.74.

[23] Gradio, ”Build & share delightful machine learning apps,” [Online]. Available: https://www. gradio.app/.