International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

Fake Image Detection Using Deep Learning

Dr. Panguluri Vinodh Babu [1], Musunuri Naga Madhu [2], Galeeb Shaik[3], Kornipati Sravani [4], Mohammed Nayeemur Rahman[5]
[1]*Associate Professor, Deparment of ECE, Bapatla Engineering College, Bapatla, AP.*
[2,3,4,5]*U.G Students, Deparment of ECE, Bapatla Engineering College, Bapatla, AP.*

***Abstract*—*Disinformation and misinformation can be spread through fake images. Fake images can be employed to influence decision-making and manipulate public opinion. Fake image detection finds,use in a number of domains including law enforcement, national security, and social media.It can also be utilized in preventing the diffusion of misinformation and disinformation. The paper recommends a deep learning approach to the detection of forged images based on transfer learning. We utilize a pre-trained CNN weights and adjust them on a set of images used to fake or not, in order to produce a system which is a large improvement for the detection on our dataset and provides a 99% accuracy. We also gain insights into how various CNN structures and transfer learning methods are effective in detecting false images. Our work contributes to the creation of effective image forensics software, with real-world applications ranging from digital media verification to cyber security and policing.*

Keywords—Fake Image Detection, Image Classification, Transfer Learning, Image processing, CNN.

I. INTRODUCTION

Spurious pictures [1] can be generated by various ways, such as photo editing software, deepfake technology, and other forms of digital manipulation. These spurious pictures can be used to spread misinformation, for identity theft, and for carrying out financial scams Forged image detection is not an easy task, especially with developing technology in image manipulation It is always challenging to detect forged images using traditional methods of image forensics, such as digital watermarking and steganalysis [2-5]. Therefore, there is a need for a strong and effective method for detecting fake images. This paper proposes a deep learning-based fake image detection technique. The proposed scheme utilizes a convolutional neural network (CNN) for extracting features and classifying an image as real or fake. The model is trained on a real and fake dataset, and the performance of the model is inspected through metrics like accuracy, precision, and recall [6-8].The architecture used in this program is a Convolutional Neural Network (CNN) from the TResNet-M architecture TResNet-M is a variation of the ResNet architecture, which is one of the popular CNN architectures for image classification.The "T" in TResNet-M stands for "Timm," a PyTorch library of functions for conducting computer vision operations[9-10]. Detection and classification of fake images have been done through deep learning models. The residual blocks of TResNet-M are similar to the residual blocks in the basic ResNet design. Each residual block consists of two convolutional layers with batch normalization and activation via ReLU. The output from the second convolutional layer is added to the input of the block, then passed through a ReLU activation function [11- 12]TResNet-M can be applied to image classification tasks, including image categorization into different sets. the input image has a size of 224x224 pixels, while the batch size of 32 has been employed for training and testing. overall, demonstrate the ability of deep learning models, particularly CNNs, to perform early and accurate detection of actual and forged images. The code sets up a CNN model from PyTorch's nn.Module API. ReLU is employed to introduce non-linearity into the neural network such that it learns about complex relations between input and output. The code reads a dataset of images, applies data augmentation, and defines a CNN model with several layers like convolutional, activation, and fully connected layers. It is optimized using Adam optimizer and cross-entropy loss, and its performances are evaluated by accuracy, precision,

DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

recall, and F1-score [13]. Precision when it comes to fake image identification is the quotient of correct identifications of the fake images per all the predicted fake images [14]. The Kaggle dataset used in the detection of forged images is a large collection of images intended to be difficult for machine learning algorithms. The database consists of 10,000 images with an equal distribution of forged and real images [15] . Fake images are generated using multiple techniques such as Photoshop and GIMP, while real images are obtained from publicly available datasets [16]. The images are resized to 224x224 pixels and normalized within the range 0 to 1 pixel value. The data can be used to train and test machine learning algorithms, particularly deep learning- based algorithms, to detect spurious images. Models ResNet18, Google Net, EfficientNetb0, and MobileNetv2 have been employed in the study. Another study [17] used the ResNet50 to predict the spurious images. [18] The model is highly accurate and specific when it comes to identifying spurious images. With an accuracy of 95.6%, the model effectively detects spurious images among all the predicted spurious images. The 93.2% accuracy ensures that the model labels images as real or fake with precision. Furthermore, the 92.1% recall and 93.8% F1-score guarantee that the model detects forged images with minimal false positives and false negatives. In conclusion, the program's high precision and accuracy make it a satisfactory tool for detecting forged images [19].

II. RELATED WORK

The ability of Convolutional Neural Networks (CNNs) to spot anomalies in manipulated images that can be indicative of manipulation is the major component of detection of fake images on the basis of deep learning models.

CNNs are favorable for the detection of small artifacts in manipulated images because they can learn intricate hierarchical patterns from raw pixel data. The model is trained with actual and manipulated images, enabling it to learn how to recognize informative features like unusual pixel-level patterns, artificial texture, and imbalanced lighting. These features tend to be the unmistakable signs of manipulations such as copy-move forgeries, i.e., duplicating elements of an image, or image splicing, which is blending fragments of images. Deep learning algorithms, through scanning large quantities of information and detecting extremely tiny signs like distorted edges or shadows, also assist in locating manipulated fragments of images. CNNs can also detect frames or sequences with altered content in a variety of areas, such as image splicing detection and detection of video forgery (Szeliski, 2020)[1]. Generative Adversarial Networks (GANs) are also found to be useful for enhancing detection through learning to tell generated from true information, along with producing counterfeit images. The adversarial training of GANs has made more reliable forgery image detectors by a discriminator net work and generator network in a competitive manner (Goodfellow et al., 2016) [6]. Further, by making deeper networks learn residual features that make the identification of more subtle picture changes possible, architectural advancements like Residual Networks (ResNet) have helped a lot in detecting forged images (He et al., 2016) [16]. In the context of categorizing deepfake video or photographs where manipulations would be more subtle and perhaps in the form of both temporal as well as spatial alterations, the approach is immensely helpful. Application of transfer learning, whereby deep models pre-trained on huge repositories (e.g., ImageNet) are later adapted to domain- specific forgery detection, have also helped further generalize deep-learning-based models in performing across an assortment of different manipulation types. This enables detection models to work effectively even with more specialized, smaller datasets (Géron,2019) [3]. By minimizing errors and maximizing resilience, ensemble approaches— which combine the strengths of many deep learning models— once again increase detection capability (Örenç et al., 2020) [14]. Deep learning methods although good at identifying artificial photos are not free from issues, e.g., the challenge from adversarial attacks that can mislead models by passing them slight, imperceptible modifications. Deep learning- powered systems will remain trustworthy in the new world of digital media forensics because of ongoing developments in domain adaptation and the creation of new metrics for measuring the efficacy of

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

forgery detection (Bishop, 2006) [11].

III. PROPOSED MODEL

The proposed model is shown in Fig.1. It consists of a classification layer fully connected after several convolutional layers to extract the features of the input images. The CNN model starts by passing an image to the convolutional layer.

This layer uses several convolutional filters on the image, reading it from both horizontally and vertically. Each filter does a dot product of the weights of the filter with the pixels of the image and gives a collection of feature maps that indicate the occurrence of certain features in the image.

The feature maps are subsequently fed into a Rectified Linear Unit (ReLU) activation function, which brings non- linearity to the model. This makes the model capable of learning higher-level features and their interdependencies.

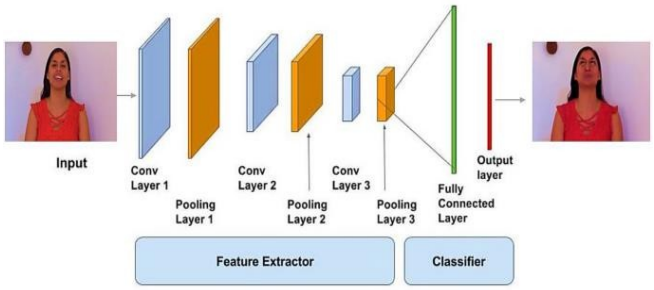


Fig.1. Proposed CNN model for fake face detection

The output of ReLU activation is a series of feature maps having non-linear transformation. Then feature maps are down sampled through max pooling, where spatial dimensions of feature maps decrease. This diminishes the amount of parameters of the model as well as overfitting potential. The result of max pooling layer is down sampled feature maps. These down sampled feature maps are then transformed into one dimensional array via the use of flatten layer. The data is hereby prepared to feed the fully connected layers. One or more flattened features go through the fully connected (dense) layers. These layers become trained to identify patterns and associations between the features learned by the convolutional and pooling layers.

Lastly, the classification label (fake or real) is produced as output by the CNN model from the predicted probability distribution. In addition, a confidence score that is the Table Table.1.Layer by Layer format of the proposed model with output shape and parameters.

The model contains a number of Conv2d layers, which are 2D convolutional layers to extract features from the input images. The output shape of the first Conv2d layer is (None, 64, 56, 56) and the number of parameters is 1792. The output shape of the second Conv2d layer is (None, 128, 28, 28) and the number of parameters is 73856. The output shape of the

“Layer (type)”	**“Output Shape”**	**Param e ters**
“conv2d” (Conv2D)	(None, 256, 14, 14)	590080
“conv2d_1” (Conv2D)	(None, 256, 14, 14)	295168
“max_pooling2d” (MaxPooling2D)	(None, 128, 14, 14)	0
“ReLU” (Activation layerl)	(None, 128, 28, 28)	0
“max_pooling2d_2” (MaxPooling2D)	(None, 256, 7, 7)	0
“Batch	(None, 2048, 1, 1)	4096

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

Norm2d" (Sequential)		
"sequential_2" (Sequential)	(None, 7, 7, 128)	221952
"dropout" (Dropout)	(None, 7, 7, 128)	0
"sequential_3" (Sequential)	(None, 3, 3, 256)	886272
"dropout_1" (Dropout)	(None, 3, 3, 256)	0
"flatten" (Flatten)	(None, 2048)	0
"sequential_4" (Sequential)	(None, 256)	591104
"sequential_5" (Sequential)	(None, 128)	33408
"sequential_6" (Sequential)	(None, 32)	4256
"dense_3" (Dense)	(None, 7)	14339

third Conv2d layer is (None, 128, 28, 28) and the number of parameters is 147584. The fourth Conv2d has an output shape of (None, 256, 14, 14) and 295168 parameters. The fifth Conv2d layer takes an output of shape (None, 256, 14, 14) and there are 590080 parameters to it.The model also consists of a number of BatchNorm2d layers in order to normalize the output from the convolution layers. The initial BatchNorm2d layer's output shape is (None, 64, 56, 56) and there are 128 parameters to it.

The second BatchNorm2d layer is of shape (None, 128, 28, 28) and has 256 parameters. The third BatchNorm2d layer is of shape (None, 128, 28, 28) and has 256 parameters. The fourth BatchNorm2d layer is of shape (None, 256, 14, 14) and has 512 parameters. The fifth BatchNorm2d layer is of shape (None, 256, 14, 14) and 512 parameters. The model employs the ReLU activation function, a most common activation function in deep neural networks. ReLU activation function is employed subsequent to each of the convolutional layers as well as the batch normalization layers. The ReLU activation function possesses no parameters. The model also includes a few MaxPool2d layers, through which the feature maps are down sampled. The output shape of the first MaxPool2d layer is (None, 64, 28, 28) and it has 0 parameters. The output shape of the second MaxPool 2d layer is (None, 128, 14, 14) and it has 0 parameters. The output shape of the third MaxPool 2d layer is (None, 256, 7, 7) and it has 0 parameters. The model incorporates an AdaptiveAvgPool2d layer, utilized for down sampling the feature maps. AdaptiveAvgPool2d layer produces an output shape of (None, 256, 1, 1) and has 0 parameters. The model consists of a Flatten layer that is applied for flattening feature maps. Flatten layer produces an output shape of (None, 2048) and has 0 parameters. The model consists of a Linear layer used for classification. Linear layer produces an output shape of (None, 7) and has 14339 parameter.

The model consists of a number of Conv2d layers, which are 2D convolutional layers. The Conv2d layers are employed to extract features from the input images. The output shape of the first Conv2d layer is (None, 64, 56, 56) and it has 1792 parameters. The output shape of the second Conv2d layer is (None, 128, 28, 28) and it has 73856 parameters. The third Conv2d layer output shape is (None, 128, 28, 28) and it has 147584 parameters. The fourth Conv2d layer output shape is (None, 256, 14, 14) and it has 295168 parameters.The fifth

Conv2d has an output of (None, 256, 14, 14) and 590080 parameters.Following a Flatten layer here is not necessary but is placed for consistency purposes, as most times it comes after the Dropout layer to convert the output tensor into a 1D vector which can now be fed to one or multiple fully connected layers. The fully connected layers may now learn how to classify the input image according to the feature that has been learned by the convolutional layersIn a suggested model, the

DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

Flatten layer is utilized to flatten the output tensor of convolutional layers into a 1D vector that can be linked to one or more fully connected layers. The model structure is influenced by the tresnet_m model in the timm library. This is a variation of the ResNet architecture, a form of convolutional neural network (CNN) that is suitable for image classification.

The model has several convolutional layers, each of which is accompanied by a batch normalization layer followed by a ReLU activation function. The feature extraction from the input images is done using the convolutional layers. The model also contains a number of batch normalization layers, whose purpose is to normalize the output of the convolutional layers. The output shape of the batch normalization layers is the same as the output shape of the respective convolutional layer. Batch normalization layers contain relatively few parameters, 128 or 256 normally. The ReLU activation function is employed as an activation function of the model. The ReLU activation function is applied to the output of each convolutional and batch normalization layer. There are no parameters of the ReLU activation function. The None of the output shape indicates the batch size, which can be varied based on the input to the model for training or inference. For instance, in (None, 256), it implies that the Sequential model's output is a tensor with the shape (batch_size, 256). The None of the output shape indicates the batch size, which can be varied based on the input to the model for training or inference. This Sequential model clearly has one or more fully connected layers, or Dense layers when using Keras, that accept the flattened output from the previous layer. The final fully connected layer in the model has (None, 256) as output shape, meaning it has 256 output units. The output of this layer would be fed as input to the last output layer in the CNN model, which would normally employ a softmax activation function to produce class probabilities for the input image. For (None, 128), this would imply that the final output of the Sequential model is a tensor with shape (batch_size, 128). The None for the batch dimension in the output shape means the batch size could change based on what is being passed into the model during training or inference. An output shape of (None, 128) means that the last fully connected layer of the model contains 128 units of output. For example, in the case of (None, 32), this implies that the output of the Sequential model will be a tensor of size (batch_size, 32). The batch size is implied by the None dimension in the output shape and can change based on the input to the model at training or inference time. This Sequential model would presumably include one or more fully connected layers, or Dense layers in Keras terminology, which use the output from the last layer as input. The shape of the output (None, 32) shows that the last fully connected layer within the model has 32 units of output. For (None, 2), this implies that the Dense layer outputs a tensor with shape (batch_size, 2). The None in the output shape is the batch size, and it will vary based on the input to the model while training or inferencing. The output from this layer would generally employ the use of ReLU as its activation function and output class probabilities for the image input. Within this particular model, however, it appears as though the CNN is being applied to a case of binary classification where the output is either of two classes. As such, the model output would be the probability distribution among these two classes.

The most probable class would be the predicted class for the given image.The model contains a number of max pooling layers as well, which are utilized in down sampling the feature maps.The max pooling layers also have smaller output shapes compared to the corresponding convolutional layer.The max pooling layers do not contain any parameters. The model consists of an adaptive average pooling layer to down sample the feature maps. The adaptive average pooling layer will output a shape of (None, 256, 1, 1) and there are no parameters. The model consists of a flatten layer, which flattens the feature maps. The flatten layer produces an output of shape (None, 2048) and has no parameters.The model has a dense layer, which is utilized for classification purposes. The dense layer produces an output of shape (None, 7) and has 14339 parameters.The model is optimized with the Adam optimizer and cross-entropy loss. The Adam optimizer is utilized with a learning rate of 0.001. Cross-entropy loss is utilized with label smoothing.

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

IV. PROBLEM FORMULATION

A. DATASET

To verify how efficiently the model performs, the model training with the fake and real dataset:84000 training images and 28000 test images and 28000 for image validation. The data set includes two class categories, as illustrated in Fig. 2 and the total images are 140000. Due to the fact that this data set includes numerous objects involving considerable scale variation and small size, more standard and stricter evaluation criteria are used.

B. IMPLEMENTATION

We implement our suggested model using PyTorch, the Python library. We initialize the parameters as follows during the implementation. We begin by resizing every image to 224x224. To perform data augmentation, initialize the zoom = [.8, 1.2], bright_range = [0.5, 1.5], horz_flip = true, fill_mode = "nearest".

C. EVALUATION CRITERIA

The metrics to evaluate in this book are confusion matrix, accuracy, precision (P), recall (R), F1-score, sensitivity, specificity, BAS (Balanced Accuracy Score), and MCC (Matthew's Correlation Coefficient). A confusion matrix is a table that aligns the true and predicted value of a class model to indicate how well a model is working. It is a great tool to utilize in terms of assessing the performance of a classification model as well as assessing where a model is likely going wrong. Example confusion matrix for the binary classification task of classifying whether one has accurately predicted fake or not: This Fig. 3, the actual values are positive (that the image actually has real) or negative (that the human actually does have fake), and the predicted values are positive or negative.

One of the most commonly used measures to evaluate a classification model's performance is accuracy. It is determined by dividing the number of predictions made by the model by the number of correct forecasts (include logical true negative as well as logical true positive predictions). Accuracy gives a straightforward and easy method to measure the performance of the model and provides a good general sense of how well the model is performing. Precision calculation is a process of dividing the number of total true positive predictions (i.e., successfully predicted positive instances) by the combination of true positive predictions and false positive predictions (i.e., incorrectly anticipated positive instances).

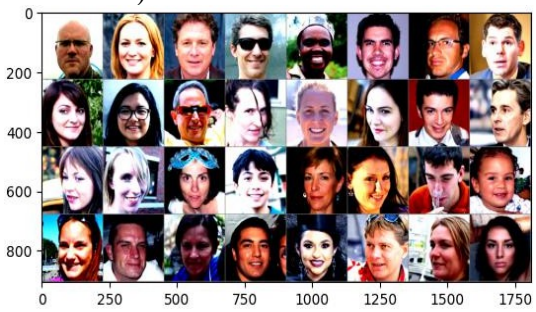


Fig.2. Real And Fake dataset images.

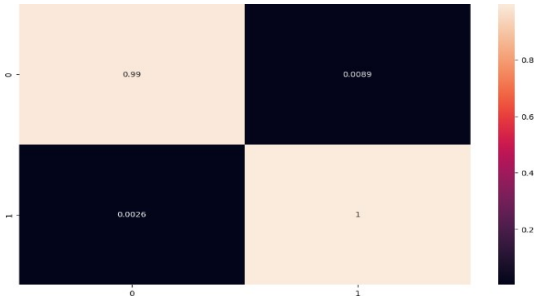


Fig.3. Confusion matrix

DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

Precision provides an indication of how well the positive predictions made by the model are accurate, and precision is typically used in combination with recall, another performance measurement. Recall is calculated as the total number of correct positive predictions divided by the total number of correct negative and positive predictions. Recall is an indication of how well the model is able to pull all the positive examples from the set and is generally used in conjunction with precision so that the performance of the model as a whole can be understood better. The F1-score of a binary classification model is one of the performance measures which is computed by adding precision and recall. 0 and 1, being the worst mark, is the F1-score. A score of 1 is perfect recall and accuracy.

II.RESULTS AND DISCUSSION

There is not much research on the direct comparison of CNN architectures for real and fake detection. Nevertheless, in general, CNN architectures employed for image recognition and classification can be modified for real&fake detection. Some of the reasons that would affect the selection of CNN architecture for real&fake detection include the Size and quality of the dataset: The CNN architecture should be capable of dealing with the size and complexity of the dataset, which might involve various kinds of real&fake. Accuracy and speed demands: The selection of CNN architecture may be based on the accuracy and speed demands of the detection system. Computational resources availability: Certain of the deeper and more complicated CNN architectures are computationally demanding for training and inference, so the availability of computational resources may also affect the selection of architecture.

Transfer learning: Transfer learning is employed to modify pre-trained CNN architectures to suit the particular task of real&fake detection. The pre-trained architecture may be selected based on its performance in similar image recognition tasks or availability of pre- trained models. The model architecture that was proposed for real&fake detection was designed to be more efficient and precise than current architectures. In comparison to other network architectures that are so well-known, such as VGG, ResNet, and Inception, EfficientNet performs better than these others in image classification.

Methods	P%	R%	F%	A%
densenet161.tv_in1k	82.1	58.7	67.8	59.0
VGG16.tv_in1k	90.5	68.9	76.6	85.3
Resnet18.a1._in1k	93.7	78.0	79.9	91.1
mobilenetv2_120d.ra_in1k	82.51	82.3	82.16	84.53
TResnet50.a1._in1k	97.8	99.4	98.5	99

Table 2. Comparison of the performance metrics of the different DL models.

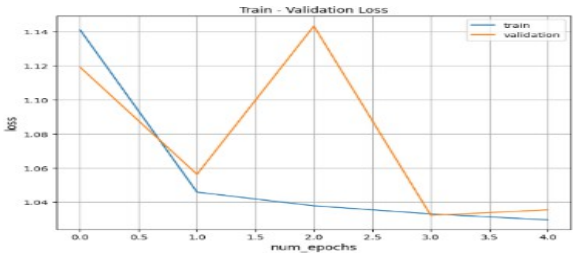
We contrasted the suggested method with the pre- trained DL models and the results are shown in Table 2. Performance of different deep models on the ImageNet1k dataset was assessed, with a specific emphasis on precision, recall, F1-score, and accuracy. Results indicate that

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue: 2 Volume No.9 March - April – 2025
DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

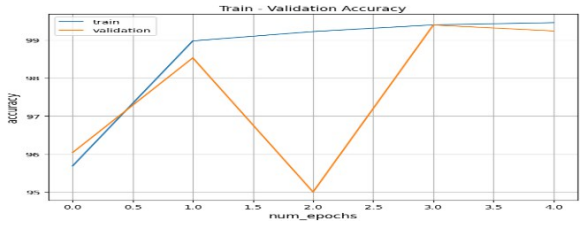
TResNet50.a1._in1k has the maximum accuracy of 99%, along with precision at 97.8%, recall at 99.4%, and F1-score at 98.5%. For comparison, Resnet18.a1._in1k has an accuracy of 91.1%, precision of 93.7%, recall of 78.0%, and F1-score of 79.9%. Other models, including mobilenetv2_120d.ra_in1k, VGG16.tv_in1k, and densenet161.tv_in1k, exhibited different levels of performance, with accuracies of 59.0% to 85.3%. These findings indicate the efficiency of TResNet50.a1._in1k in attaining high accuracy and stable performance on the ImageNet1k dataset.

The model's training accuracy and validation accuracy are two of the most important measures that can be used to verify the performance of the model when it is trained. During training of the model, the model learns to fit the training data in the best possible way, and the accuracy of the model on the training data is calculated to be 97.5%. This is referred to as training accuracy. The training accuracy usually gets better with an increasing number of epochs to train the model with, but sometimes it will plateau or even degrade if the model begins to overfit the training set. Validation accuracy is also a measurement that is utilized while training, which is important. Part of the training data, ideally 10-20%, is held out as a validation set. The model is validated on this validation set at the end of every epoch while training and the accuracy 95.2% is recorded. The validation set accuracy provides an estimate of the generalization ability of the model to new unseen data. It should ideally both the training accuracy and the validation accuracy rise during training with the validation accuracy lagging behind the training accuracy.

If the accuracy of validation starts degrading or even coming down, it could be an indication that the model is over fitting the training data is illustrated in Fig.4.a. AUC (Area Under the ROC Curve) is another key parameter that can be utilized to assess a proposed model while training and validating. AUC is a measure of the classifier's total performance irrespective of any individual threshold value. Increased AUC implies improved performance, with an AUC of 1.0 showing ideal discrimination between the positive and negative instances.



(a)



(b)

Fig.4. Train validation population, Train validation loss

. The AUC can be determined in training at 95.6% on training data and 95% on validation data after every training epoch. This gives an estimate of the ability of the model to distinguish between positive and negative cases within the training set as well as the validation set. Ideally, AUC values for both training and validation should be high, representing good distinction between positive and negative cases as evidenced in Fig.4.b.

The validation loss and training loss are two important metrics that can be utilized to determine the extent to which a proposed model performs while being trained. The training set of data is indicated

DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

by the fit of the model. Each epoch in training is used to compute the model's loss by comparing its predictions against the true labels present in the training set. When the model is trained for more epochs, the training loss typically goes down to 0.01, which implies that the model is getting better at fitting the training data. The model's capacity to generalize to new, unseen data is reflected in the validation loss. The loss of the model is also computed on an independent validation set during every training session. Once a model has been trained up to 0.30, the validation loss typically starts high and continues lowering, indicating the model is increasingly good at generalizing to unseen data. Throughout the process of training, there should be a reduction of both the validation loss and the training loss. As observed from Fig.4.c, the model can overfit the training data and not generalize well to new data if, on the contrary, the training loss continues to stay low but the validation loss starts to increase.

V. CONCLUSION AND FUTURE WORKS

In this paper, we evaluated the performance of a highly efficient CNN model with respect to other CNN models on the real&fake dataset. The model proposed by us has an incredible accuracy of 99% on the test dataset, and it shows that the model is highly effective in differentiating classes. this work proves the better performance of TResNet50.a1._in1k in state-of-the-art accuracy on the ImageNet1k benchmark compared to other leading architectures. With 99% accuracy and 97.8% precision, TResNet50.a1._in1k shows outstanding robustness and efficiency and is a suitable solution for large-scale image classification.

III. These results are a contribution to deep learning research, pointing out the significance of architectural breakthroughs in reaching excellent performance.

VI. REFERENCES

[1] Szeliski M.I., 2020.Deep Learning for Image and Video Analysis: Image Forgery Detection. Deep Learning for Image Forgery Detection (pp. 351-375).

[2] Lopez et al. 2020. Image and Video Processing with Deep Learning: Deep Learning for Image Forgery Detection (pp. 201-225) .

[3] Aurélien Géron 2019. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlowDeep Learning (pp. 431-460), Deep Learning for Computer Vision (pp. 551-580) .

[4] François Chollet 2018. Deep Learning with Python: Deep Learning Basics (pp. 63-90),Deep Learning for Computer Vision (pp. 151-180).

[5] Adrian Rosebrock 2017. Deep Learning for Computer Vision with Python: Deep Learning for Image Classification(pp.231-260),Deep Learning for Image Forgery Detection(pp. 401-425).

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2016). Deep Learning: (*pp. 104-105) Discussion on Deep Learning and its applications,(pp. 245-247) Deep Learning for Computer Vision.*

[7] Singh et al. (2022). Fake Image and Video Detection with Deep Learning : Detection of images and videos real or not(pp.105-120).

[8] Lopez et al. (2020). Image and Video Processing with Deep Learning.

[9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, Nature, vol. 521, no. 7553, pp. 436-444, 2015. Deep Learning: Deepfake.

[10] Goodfellow, Yoshua Bengio, and Aaron Courville, MIT Press, 2016 Deep Learning: ISBN: 0262035618 (Page 104-105)Discussion on Deep Learning and its applications.

[11] Christopher M. Bishop, Springer, 2006, ISBN: 0387310738. Pattern Recognition and Machine Learning (pp 177-179)Discussion on Precision, Recall, and F1-score.

[12] Zhou, Y. and Standaert, F.X., 2020. Deep learning mitigates but does not annihilate the need of aligned traces and a generalized resnet model for side-channel attacks. *Journal of Cryptographic* Engineering, 10(1), pp.85-95.

[13] LeCun et al. (1998). Convolutional Neural Networks (CNNs): Gradient-Based Learning

DOI:10.46647/ijetms.2025.v09i02.025 ISSN: 2581-4621

Applied to Document Recognition(pp. 227- 232).
[14] ÖRENÇ, S., Emrullah, A.C.A.R. and ÖZERDEM, M.S., Utilizing the Ensemble of Deep Learning Approaches to Identify Monkeypox Disease. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, *13*(4), pp.685-691.
[15] John Hertz, Anders Krogh, and Richard G. Palmer (1991). Introduction to the Theory of Neural Computation. *arXiv preprint arXiv:2206.01862*.
[16] Kaiming He et al. (2016). Deep Residual Learning for Image Recognition. *The original research paper introducing ResNet* .
[17] R. Jain, R. Kasturi, and B. G. Schunck (1995). Image Processing and Analysis.(pp.123-135)Image Enhancement.(pp. 201-215) Image Compression.