# Image Forensics

***Abstract*— In this research endeavour, we address the pressing concern of digital image forgery detection by proposing two novel algorithms grounded in distinct methodologies. The first algorithm leverages the logarithmic basis of Benford's law, asserting uniform distribution of mantissa in the logarithm of practical numbers. Our approach utilizes this principle to analyze the mantissa distribution of discrete cosine transform (DCT) coefficients in JPEG images, employing a support vector machine (SVM) for classification. The results demonstrate exceptional accuracy in CASIA V1.0 and V2.0 datasets. The second algorithm tackles the challenge of unified detection for splicing and copy-move forgery. By exploiting artifacts from manipulations in JPEGencoded images, we assess changes in the statistical properties of AC components of DCT coefficients. The proposed technique incorporates standard deviation and the count of non-zero DCT coefficients for each AC frequency component independently. Experimental validation on the CASIA V1.0 and V2.0 datasets, both pre-and post-processed, showcases superior detection rates. Through a series of rigorous experiments, we unveil the limitations of these models, showcasing their struggles in cross-dataset evaluations and when confronted with inthe-wild manipulated media. The findings spotlight the discrepancy between reported high performance on specific datasets and the actual performance in diverse, real-world scenarios. This research prompts a reconsideration of evaluation methodologies. It highlights the need for more resilient and adaptable image forgery detection techniques in the face of evolving digital manipulation techniques and varied data distributions.**

***Keywords—mantissa, DCT, Benford's, forgery, coefficient***

## I. Introduction

Digital image forgery has emerged as a pervasive concern in today's technologically advanced and interconnected world. The ability to manipulate visual content, fueled by sophisticated tools and widespread dissemination through large-scale social platforms, has raised profound implications for the authenticity of digital media. The consequences of undetected image manipulation are farreaching, encompassing misinformation, privacy breaches, and the erosion of trust in visual information. As a result, the development of robust and effective techniques for image forgery detection has become a paramount research focus.

*Image Splicing*



This study delves into the complex domain of detecting image forgery, seeking to offer fresh perspectives and methodologies to tackle the dynamic realm of digital manipulation. Image forgery involves more than surfacelevel visual changes; it encompasses a range of techniques, including splicing, copy-move, and more advanced manipulations. Confronting these challenges requires not just an appreciation for the nuances of digital manipulation but also the creation of techniques adept at identifying various types of forgery.

While significant strides have been made in the field of image forgery detection, there exist inherent challenges, notably the need for approaches that demonstrate robust performance across diverse datasets and real-world scenarios. This introduction sets the stage for our research, which encompasses two distinctive yet interconnected approaches. The first proposes an innovative forgery detection algorithm grounded in the logarithmic basis of Benford's law, while the second tackles the unified detection of splicing and copymove forgery through a meticulous analysis of JPEGencoded images. Furthermore, recognizing the limitations of current state-of-the-art techniques, we extend our investigation to scrutinize the generalizability of existing detection methods, particularly those reliant on handcrafted features. This evaluative examination seeks to illuminate performance gaps and underscores the imperative for adaptive methodologies capable of navigating the complexities presented by the ever-changing landscape of digital manipulation.

*Authentic Image (left) and Copy-Move (right) Image Forgery*

In the following sections, we explore the details of each proposed approach, providing methodologies, experimental outcomes, and a comparative assessment against established methods. Through this investigation, our goal is to not only advance image forgery detection techniques but also to enrich the overall comprehension of the field's existing constraints and identify potential paths for future exploration and potential avenues for future exploration.

## II. Main Contribution

Our research contributes significantly to the domain of image forgery detection through the development of a novel methodology that encompasses feature extraction and the construction of a robust classification model tailored specifically for colour images. Distinguishing itself from existing techniques, our approach intricately utilizes the

Discrete Cosine Transform (DCT) as a central element for feature extraction, fostering a comprehensive understanding of image intricacies. The innovation lies in the comparison of various techniques employing the DCT approach, offering a nuanced examination of their efficacy in forgery detection. Moreover, we conduct a thorough exploration of classifiers, delving into Support Vector Machine (SVM) and Random Forest to discern their respective merits in this context. The application of these classifiers is rigorously benchmarked against two prominent datasets, CASIA 1.0 and CASIA 2.0, facilitating a comprehensive evaluation of our proposed methodology.

To enhance the efficiency of our model, we introduce optimization techniques, including Principal Component Analysis (PCA), Kfold cross-validation, and Bayesian optimization. The meticulous experimentation process is carried out on Google Colab GPU, ensuring accelerated processing speeds, and facilitating a seamless exploration of the model's capabilities. In essence, our contribution extends beyond the mere development of a forgery detection technique; it encompasses a thorough examination of key parameters, classifiers, and optimization strategies, positioning our work as a valuable resource for advancing the state-of-the-art in image forgery detection.

## III. Related work

In recent times, numerous innovative approaches have emerged for passive detection of both copy–move and image splicing forgeries. Shi and Chen's method [1] stands out by modeling tampering changes through statistical features derived from 2D arrays generated via multi-size block discrete cosine transform (MBDCT). Achieving an impressive 91% accuracy on the Columbia Image Splicing Detection Evaluation Dataset (Columbia) [2] with SVM showcases its efficacy. Another noteworthy technique by Zhen and Jiquan [3] characterizes tampering changes through moment features extracted from 2D arrays generated by applying MBDCT and image quality metrics (IQMs). This method, utilizing SVM for classification, attains an accuracy of 87.10% on the Columbia dataset. Li et al. [4] propose an innovative image splicing detection technique based on Markov features in the Quaternion discrete cosine transform (QDCT) domain. The integration of QDCT aims to leverage the entire color information, with expanded Markov features extracted from intra-block and inter-block QDCT coefficients matrices. Wei and Jing [5] adopt a unique approach by modeling tampering changes through the stationary distribution of the edge image extracted from the chroma component, utilizing a finite-state Markov chain. Employing SVM as a classifier, this method achieves an impressive accuracy of 95.6% on CASIA v2.0. Fridrich et al. [6] introduced a pioneering technique for detecting copymove forgery (cmf) using features derived from DCT coefficients in small overlapping image blocks. However, challenges arose when applied to images with large identical textured regions, leading to false matches. Prakash et al. [7] proposed an integrated technique for detecting both splicing and cmf. Their study introduces an enhanced threshold method based on the Markov random process to extract features from different color spaces. Notably, the scheme remains unexplored with a combined collection of authentic, spliced, and cmf images for both datasets.

## IV. Methodology

### *A. Image forgery detection based on Discrete cosine transform (DCT) coefficients*

Discrete Cosine Transform (DCT) is a mathematical transformation commonly used in image processing and JPEG compression. It converts spatial information such as an image block into frequency components. These components are also called coefficients. For example, if we apply DCT to an 8x8 image block of pixels, then the output will be an 8x8 block of frequency coefficients. So, now in the output each pixel is represented by its respective frequency component/coefficient. Out of a total of 64 coefficients, the first coefficient is DC (direct current) and the rest of the 63 are AC (alternate current) coefficients.

Below is the mathematical formula to calculate a DCT value for a pixel in an n x n block.

$$F(u,v) = C(u) \cdot C(v) \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cdot \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cdot \cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

Here:

- $F(u,v)$ is the DCT coefficient at position $(u,v)$.
- $f(x,y)$ is the pixel value at position $(x,y)$.
- $C(u)$ and $C(v)$ are normalization constants, defined as 0.707 for $u=v=0$ and 0.5 otherwise.
- $N$ is the size of the image block (number of pixels along one dimension).

In Python, we have an inbuild function to calculate the DCT coefficients of an image block. The function is given as:

dct_block = dct(block, type=2, norm=None)

The type=2 argument specifies the Type-II DCT. The TypeII DCT is commonly used in image compression and related applications.

In our work, we are using Discrete Cosine Transform (DCT) coefficients to detect the forgeries in an image. When a JPEG-compressed image is tampered with, there is a change in the statistical properties of AC components of block DCT coefficients. So, we capture these changes by calculating the standard deviation and count of non-zero DCT coefficients corresponding to the respective AC coefficients across all DCT blocks. Then use these changes to identify a forged image. Below Fig.1a represents an E2E workflow of this technique.

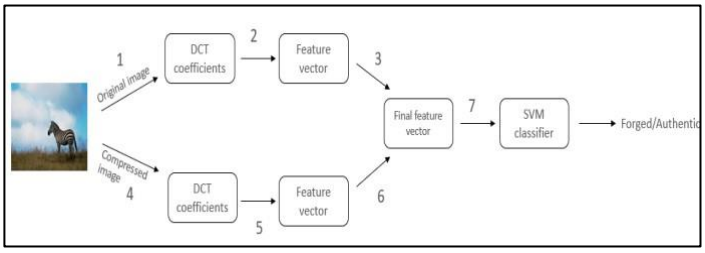



*Fig.1a*

In this, we have numbers representing the flow sequence of the technique. The step-by-step working of the technique based on the above workflow is as follows.

1. Suppose we have an image. First, the original image is divided into 8x8 blocks of pixels. So, now each block contains in total of 64 pixels.

2. Next, for each 8x8 block of pixels, DCT coefficients are calculated. After the calculation, the output has the same number of blocks. But now each 8x8 block

contains 64 DCT coefficients for the respective pixels. The first DCT coefficient in each block is a DC and the rest 63 are AC. We need to work with AC coefficients. So, eliminate the first DCT coefficient from each block. Now each block has 63 coefficients belonging to AC. Now, take the first block of DCT coefficients of an image and arrange its DCT coefficients in row row-wise manner (63x1). So, now we have the first column with 63 rows. Then do the same for the second block of an image. Now, we have 2 columns and 63 rows. Next, do the same for the rest of the blocks of an image. Finally, we have 63 rows and n columns. Now, calculate the standard deviation of each row. After calculating the standard deviation for each row, now we have in total 63 values of standard deviation corresponding to the 63 rows in one column. Now arrange these 63 values of standard deviation in row row-wise manner (1x63).

Now, count the non-zero values for each row (63xn). After counting non-zero values for each row, again we have 63 values of non-zero count corresponding to the 63 rows. Now arrange these 63 values also in row row-wise manner (1x63). And append this same row of non-zero count to the standard deviation row. Now, we have in total 126 values (63 for standard deviation + 63 for non-zero count) in a row.

3. Now, add these 126 values in the same row-wise manner (1x126) to the Final feature vector.
4. Now, cropped the original image by removing its 4 rows and 4 columns from the top left corner to simulate the compression of an image. Then repeat the steps 1-4 for this compressed version.
5. Now, our final feature vector has in total of 256 values (1x256). First 126 values for the original image and next 126 for its compressed version.
6. Repeat steps 1-4 for the other two channels (Cr and Cb) of an image. Then, finally, we have in total of 756 values in our final feature vector for an image.
7. Then we put this feature vector (1x756) corresponding to an image to the machine learning classifier (SVM/Random_Forest) for the classification. Then the machine learning classifier will tell us if an image is forged or not.

*B. Image forgery detection based on Mantissa Distribution*

Benford's Law, also known as the first-digit law, posits that in many naturally occurring datasets, the distribution of the first digits of numerical values is not uniform but follows a logarithmic pattern. Specifically, the probability P(d) of the first digit being d (where d ranges from 1 to 9) is given by:

$$P(d) = \log_{10}(d+1) - \log_{10}(d) = \log_{10}\left(\frac{d+1}{d}\right) = \log_{10}\left(1+\frac{1}{d}\right)$$

$$\Phi(x) = \log_{10} x \mod 1$$

$$\Phi(x) \sim \text{Uniform}\,[1, 0)$$

The quantity is proportional to the space between *d* and *d* + 1 on a logarithmic scale. Therefore, this is the distribution expected if the *logarithms* of the numbers (but not the numbers themselves) are uniformly distributed.

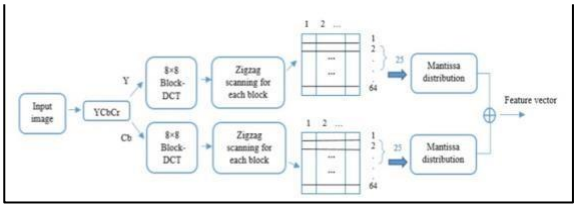
This principle has found applications in various fields, including fraud detection, where deviations from the expected distribution may signal anomalous behaviour. In our context, we leverage Benford's Law to scrutinize the mantissa distribution of the logarithms of DCT coefficients in JPEG images, seeking deviations that may indicate image forgery. The logarithmic basis of Benford's Law aligns with the inherent characteristics of natural data. In our approach, we consider the mantissa of the logarithm of practical numbers extracted from DCT coefficients. Given that the mantissa should follow a uniform distribution according to Benford's Law, any significant deviation suggests potential forgery.

Before applying Benford's Law to our dataset, we preprocess the JPEG images to extract the DCT coefficients. The images are divided into non-overlapping blocks of size 8x8 pixels, and DCT coefficients are computed for each block independently. The mantissa of the logarithm is then extracted from these coefficients. Then, for each image, the feature extraction process involves capturing the distribution of mantissa from the logarithms of DCT coefficients. This forms a feature vector representing the unique characteristics of the image under scrutiny. The feature vector is a quantitative representation of how closely the mantissa distribution adheres to the expected pattern from Benford's Law. To discern authenticity from forged images, we employ a Support Vector Machine (SVM) for classification. The SVM is trained on a labelled dataset, learning to differentiate between the mantissa distributions corresponding to authentic and forged images. This classification model is then applied to new, unseen images to predict their authenticity based on the learned features. The effectiveness of our methodology is evaluated using standard performance metrics, including accuracy. These metrics quantify the algorithm's ability to correctly classify authentic and forged images. The experiments are conducted on the CASIA V1.0 and V2.0 datasets, and the results are compared with other methods to establish the superiority of our proposed forgery detection algorithm.

In our approach to image forgery detection, we adopt a meticulous strategy focused on the luminance (Y) and chrominance (Cb and Cr) components. The decision to leverage Y and Cb channels is rooted in their demonstrated superior performance and the advantage of reduced feature dimensions in our model. Consequently, our methodology unfolds in a series of steps for each channel.

Initially, the image is subjected to division into nonoverlapping 8x8 blocks, laying the groundwork for subsequent analysis. A pivotal aspect of our technique involves the application of a two-dimensional discrete cosine transform (2D-DCT) to each block, resulting in the extraction of 64 discrete cosine transform coefficients (DCT coefficients). To streamline the management of these coefficients, we employ zigzag scanning for each block, orchestrating an ordering of DCT coefficients from high to low frequency. The upper-left corner of the block harbours the direct current (DC) coefficient, while the remaining 63 values constitute alternate (AC) coefficients.

The subsequent step entails the construction of a zigzag matrix, wherein vectors obtained from the previous zigzag scanning are arranged in columns to form a matrix with 64 rows and columns, mirroring the number of image blocks. Noteworthy studies (e.g., [11, 18, 19, 20, 21]) have highlighted the impactful role of low-frequency AC components in the efficacy of forgery detection systems. Motivated by these findings, we selectively consider the first 25 AC modes in our algorithm, resulting in the preparation of 25 vectors for subsequent processing.




An additional layer of sophistication is introduced as we apply the mantissa distribution to each vector obtained from the previous step. This involves generating histograms of the mantissa for each mode, a process repeated for both Y and Cb channels. Consequently, the 25 output vectors from each channel are amalgamated into a singular vector, yielding two comprehensive output vectors for further analysis.

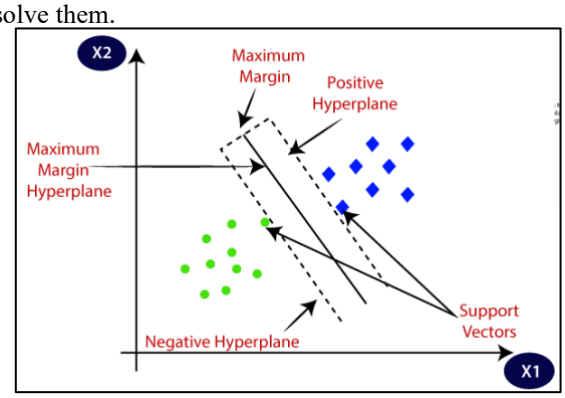
The culmination of our methodology involves the extraction of a feature vector, wherein the two output vectors (representing Y and Cb channels) are combined. This amalgamated feature vector encapsulates 500 features (2×25×10), providing a rich representation of the image's intricacies. To discern between authentic and forged images, we employ the Support Vector Machine (SVM) for classification. Our experimentation extends to the exploration of three types of SVM kernel functions—linear, quadratic, and cubic—the outcomes of which are meticulously detailed in Section III of this paper, offering valuable insights into their respective performances and contributions to the effectiveness of our forgery detection approach.

### *C. Support Vector Machines (SVM)*

This section explains the working of the SVM algorithm which we are using with our techniques for the classification of images (Authentic/forged). The description of the SVM is as follows.

SVM is a supervised machine learning problem where we try to find a hyperplane that best separates the two classes. It can be used for both regression and classification tasks, but generally, they work best in classification problems. There are 2 types of Support Vector Machine Algorithms linear SVM and non-linear SVM.

Linear SVM we used when the data is perfectly linearly separable. When data points are perfectly linearly separable, only one straight line is needed to divide them into two classes. Non-linear SVM we used when the data is not linearly separable, which means when the data points cannot be separated into 2 classes by using a straight line, then we classify them using some sophisticated methods like kernel trickery. In most real-world applications we do not find linearly separable data points hence we use kernel tricks to solve them.




There are two classes in our dataset (green and blue). SVM has separated the 2 classes using a maximum margin hyperplane. Hyperplanes, which are decision boundaries, help classify the data points. The classes of the data points that lie on either side of the hyperplane are distinct. The number of features also affects the hyperplane's dimension. When input features are limited to two, the hyperplane can be thought of as a line. The hyperplane transforms into a twodimensional plane if there are three input features. To classify these points, we can have many decision boundaries, but the best hyperplane is that plane that has the maximum distance from both classes, and this is the main aim of SVM. The data points that are near by the hyperplane are called as support vectors. Because they establish the margin, these are the most crucial points for the SVM. By utilizing these support vectors, we are able to increase the classifier's margin. In SVM the large margin is considered a good margin. Now, if any new point will come, then SVM will classify that point accordingly. To build this hyperplane to separate the classes, SVM uses Kernel Tricks. The choice of kernel in an SVM is crucial as it determines the form of the decision boundary. Here are some commonly used SVM kernels.

#### *a) Linear Kernel*

The linear kernel is the most straightforward. It represents the input data in the same feature space where the classes are most separable by a hyperplane. It is suitable for linearly separable data. Below is the Python function representing the linear kernel:

```
model_SVC_poly = SVC(kernel='linear')
```

#### *b) Polynomial Kernel*

The polynomial kernel introduces non-linearity by mapping the input features into a higher-dimensional space using polynomials. It is controlled by a degree parameter that determines the order of the polynomial. 'C' determines the margin distance between the support vectors of different classes. Below is the Python function representing the polynomial kernel:

```
model_SVC_poly = SVC(kernel='poly', degree=3, C=1)
```

#### *c) Radial Basis Function (RBF) Kernel*

The RBF kernel, also known as the Gaussian kernel, is widely used for non-linear classification. It maps the input data into an infinite-dimensional space, allowing the SVM to capture complex decision boundaries. Controlled by a parameter called gamma, which influences the shape of the decision boundary. Below is the Python function representing the RBF kernel.

```
model_SVC=SVC(kernel='rbf',C=100,gamma=0.001)
```

## V. Experiment and Results

All the experiments have been performed on Google Colab's GPU for faster processing. This helped us to optimize the execution time/time complexity taken by the sets of experiments on the CPU. For the experimentation, we have used two different datasets [8] **CASIA 1.0** and **CASIA 2.0**. CASIA 1.0 has a total of 1720 color images. Out of 1720, 800 are Authentic images 460 images are Copy-move forged and 460 are Splicing Forged. All images are in .jpg format with a size of 384 x 256. CASIA 2.0 has around 2000 colour images. Out of 2000, around 850 are authentic images 560 are Copymove forged and 560 are Splicing forged. All images are in .jpg format with varying sizes.

### *A. Discete cosine tranform (DCT) coefficient*

This section contains the experimental results given by this technique (DCT coefficient). As we have used two datasets CASIA 1.0 and CASIA 2.0. In each section (*1) Using CASIA 1.0 and 2) Using CASIA 2.0*) below, first we

have used the DCT coefficient technique to extract the features from the images and put them into an Excel file with respective labels (0 or 1). '0' for authentic images and '1' for forged/tampered images. Then, we will use that Excel file to train our machine learning model (SVM/Random Forest) after splitting the samples inside the Excel file into training (70%) and testing sets (30%). Then, at last, using that trained model (SVM/Random Forest) to perform the classification on the testing set.

*1) Using CASIA 1.0:*

This section contains the results from the CASIA 1.0 dataset. Using this dataset, we have performed multiple sets of experiments to validate the robustness of techniques across different forgeries. The experiments using the CASIA 1.0 dataset are as follows.

*a) First experiment – This part contains the sets of experiments using SVM as a machine learning algorithm for the classification.*

For forged images, we have both the categories Copymove and Splicing. In total, we have 1720 images. Out of which 800 are Authentic images, 460 images are Copy-move forged and 460 are Splicing Forged. After extracting the features using DCT, we have in total of 1720 samples in our Excel file. We split these samples into training (70%) and testing (30%) sets. For the classification, we have chosen a support vector machine (SVM) algorithm with default parameters i.e. we have not done any optimization as of now.

```
Results before Optimization:
Accuracy: 0.8568665377176016
Confusion Matrix:
 [[194  46]
 [ 28 249]]
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.81      0.84       240
           1       0.84      0.90      0.87       277

    accuracy                           0.86       517
   macro avg       0.86      0.85      0.86       517
weighted avg       0.86      0.86      0.86       517
```

*Fig.3a* Results before optimization.

In Fig.3a, we can see the accuracy is 85%. For the testing/validation set, we have in total of 517 samples. Out of which 240 are authentic and 277 are forged/tempered (mix of Copy-move and Splicing). From the confusion matrix, we can see that out of 240 authentic, the model has correctly classified 194 as authentic while misclassified 46 as forged. Out of 277 forged, the model has correctly classified 249 as forged and misclassified 28 as authentic.

Now, to increase the accuracy of our model we have performed the optimization by applying Principal component analysis, Bayesian optimization and K-fold cross-validation. The principal component analysis we have applied to our data set for the dimensionality reduction to avoid overfitting. Bayesian optimization we have applied for the hyperparameter tuning which helped estimate the best set of parameters for our model. K-Fold cross-validation helped build a robust model for the classification by validating the model across different folds of training and validation sets. After applying these optimization techniques, we trained our model and performed the validation using the same testing/validation set. Below are results in Fig. 4a showing the accuracy, confusion matrix and classification report for the same.

```
0.9632495164410058
[[227  13]
 [  6 271]]
              precision    recall  f1-score   support

           0       0.97      0.95      0.96       240
           1       0.95      0.98      0.97       277

    accuracy                           0.96       517
   macro avg       0.96      0.96      0.96       517
weighted avg       0.96      0.96      0.96       517
```

*Fig.4a Results after optimization.*

We can see that accuracy increased to 96%. This model has correctly classified 227 as authentic out of 240 and 271 as forged out of 277. So, the optimization techniques have effectively enhanced the accuracy of the model from 85% to 96%.

Next, we have checked the robustness of this technique for a particular type of forgery (Copy-move or splice). This helps to validate whether the model is biased or not toward any forgery (Copy-move or splice). So, first, we have validated our technique for Copy-move forgery only. To do this, we have excluded all the Splicing images from our dataset. Now, our dataset contains in total of 1260 images. Out of which 800 are authentic and 460 are Copy-move forged.

```
0.9735449735449735
[[234   6]
 [  4 134]]
              precision    recall  f1-score   support

           0       0.98      0.97      0.98       240
           1       0.96      0.97      0.96       138

    accuracy                           0.97       378
   macro avg       0.97      0.97      0.97       378
weighted avg       0.97      0.97      0.97       378
```

*Fig. 5a Copy-move forgery.*

Then we performed the feature extraction using the DCT coefficient's technique on this dataset and put the extracted features for each image in an Excel file (CASIA1_Copy_move_features.csv). Now our Excel file has in total of 1260 samples. Then, we split these samples into training (70%) and testing (30%) sets. Using the optimized version of our SVM model for the classification, below Fig.5a shows the results for the same.

We can see that accuracy is 97%. Out of 138 copy-forged samples, the model has classified 134 correctly. And out of 240 authentic, it has classified 234 correctly.

Next, we have validated our technique for Splicing forgery only. Now, we have excluded all the Copy-move forged images from our dataset. So, the resulting dataset contains in total of 1260 images (800 are authentic and 460 are Splicing forged). With the same procedure and set of configurations, as in the previous sub-experiment (Fig. 3), we have performed the classification on this dataset. Below Fig.6a contains the result for the same.

```
0.9762532981530343
[[237   3]
 [  6 133]]
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       240
           1       0.98      0.96      0.97       139

    accuracy                           0.98       379
   macro avg       0.98      0.97      0.97       379
weighted avg       0.98      0.98      0.98       379
```

*Fig. 6a Splicing forgery.*

As shown in Fig. 6a, accuracy is 98%. From the results shown, we can conclude that the DCT coefficient technique with the SVM classifier is robust across Copy-move and Splicing forgery.

*b) Second experiment – In this, we are testing the DCT coefficient technique with a Random Forest Classifier instead of a Support Vector Machine (SVM). The purpose is to check if Random Forest surpasses the SVM or not.*

In this experiment, we have performed a Random Forest algorithm for the classification instead of a Support Vector Machine (SVM). The rest of the experimental setup is the same. So, first, we performed experimentation on the dataset which contains both Copy-move and Splice types of forged images in addition to authentic images. The experimental settings are the same as in Fig.4a except the classifier i.e. Random Forest in this case. Below Fig.7b shows the confusion matrix and classification report for the same.

```
[[200  40]
 [ 41 236]]
              precision    recall  f1-score   support

           0       0.83      0.83      0.83       240
           1       0.86      0.85      0.85       277

    accuracy                           0.84       517
   macro avg       0.84      0.84      0.84       517
weighted avg       0.84      0.84      0.84       517
```

*Fig.7b Random Forest (mix forged images)*

In Fig.5, we can see that even after applying the optimization techniques, the random forest has given an accuracy of 85% which is less than SVM.

Next, again with the same experimental setups, as in Fig.5a Copy-move and Fig.6a Splicing., we performed the validation of the DCT technique with the Random Forest algorithm on these two cases individually. The purpose is to check whether Random Forest is biased or not toward any forgery (Copy-move or Splicing). After performing the classification, the below Fig.8b (Copy-move) and Fig.9b (Splicing) show the results in classification reports.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.98      0.89       240
           1       0.95      0.61      0.74       138

    accuracy                           0.85       378
   macro avg       0.88      0.80      0.82       378
weighted avg       0.87      0.85      0.84       378
```

*Fig.8b Random Forest (Copy-move)*

```
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.95      0.92       240
           1       0.90      0.81      0.85       139

    accuracy                           0.90       379
   macro avg       0.90      0.88      0.89       379
weighted avg       0.90      0.90      0.90       379
```

*Fig.9b Random Forest (Splicing)*

In Fig.8b and Fig.9b, we can see that the accuracy for Copy-move is 85% and for Splicing is 90%. For Splicing the accuracy increased to 90% but still, it is less than SVM. So, from the results, we can conclude that the Random Forest classifier is less accurate than SVM. SVM is better for image forgery detection. So, now we will use the SVM classifier only in the next set of experiments.

*2) Using CASIA 2.0: The experiments under this section are based on dataset CASIA 2.0.*

The purpose of this experiment, using CASIA 2.0, is to check whether the DCT coefficient technique with SVM is robust across diverse datasets having different kinds of authentic, Copy-move and Splicing images.

So, in this we have repeated the same sets of experiments using SVM as shown in Fig.4a, Fig.5a and Fig.6a. All the experimental settings are the same except the dataset. In this, we are using CASIA 2.0 instead of CASIA 1.0. So, below Fig.10b, Fig.11b and Fig.12b show the results based on CASIA 2.0.

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       253
           1       0.99      1.00      0.99       336

    accuracy                           0.99       589
   macro avg       0.99      0.99      0.99       589
weighted avg       0.99      0.99      0.99       589
```

*Fig.10b. Mixed (Copy-move and Splicing) images*

```
0.9929078014184397
[[251   2]
 [  1 169]]
              precision    recall  f1-score   support

           0       1.00      0.99      0.99       253
           1       0.99      0.99      0.99       170

    accuracy                           0.99       423
   macro avg       0.99      0.99      0.99       423
weighted avg       0.99      0.99      0.99       423
```

*Fig.11b Copy-move forgery.*

```
0.9952267303102625
[[251   2]
 [  0 166]]
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       253
           1       0.99      1.00      0.99       166

    accuracy                           1.00       419
   macro avg       0.99      1.00      1.00       419
weighted avg       1.00      1.00      1.00       419
```

*Fig.12b Splicing forgery.*

In Fig.10b, Fig.11b and Fig.12b, we can see that validation accuracy is the same in each of the cases which is 99%. From these experimental results, we can conclude that the DCT coefficient technique with an SVM classifier is robust across different types of forgeries in CASIA 2.0.

When experimented with the Mantissa distribution the dataset CASIA 1.0 showed an accuracy of 96% after optimization as shown in Fig13b. However, the same algorithm produced a result which was not expected from the updated dataset of pictures using CASIA2.0 and has shown accuracy lesser than the CASIA 1.0 dataset. The CASIA 2.0 results are shown in Fig.14b which was obtained after the optimization.

```
0.965183752417795
[[253  12]
 [  6 246]]
              precision    recall  f1-score   support

           0       0.98      0.95      0.97       265
           1       0.95      0.98      0.96       252

    accuracy                           0.97       517
   macro avg       0.97      0.97      0.97       517
weighted avg       0.97      0.97      0.97       517
```

*Fig 13b CASIA1.0 for Mantissa Distribution*

```
0.9354838709677419
[[266   3]
 [ 35 285]]
              precision    recall  f1-score   support

           0       0.88      0.99      0.93       269
           1       0.99      0.89      0.94       320

    accuracy                           0.94       589
   macro avg       0.94      0.94      0.94       589
weighted avg       0.94      0.94      0.94       589
```

*Fig 14b CASIA2.0 for Mantissa Distribution*

These results have shown that Methods report high accuracy on the datasets that the models are trained on but fail to perform well on test datasets that are slightly different in distribution. Also, these results show that the efficacy of current algorithms is subject to potential challenges in the future, as advancements in editing software tools may render them less effective or obsolete. The dynamic nature of technology necessitates continuous adaptation and refinement to ensure sustained performance.

### *B. Results*

As shown in Fig.13b, in the case of CASIA1.0, the Mantissa distribution technique's accuracy is kind of like the DCT coefficient technique's average accuracy for mixed (Copy-move + Splicing) and Splicing forgery i.e. around 97%. But in the case of Copy-move forgery, the DCT coefficient technique has shown higher accuracy i.e. 97% than the Mantissa distribution technique.

| Technique | Foregery type | Accuracy % | |
|---|---|---|---|
| | | CASIA 1.0 | CASIA 2.0 |
| Discrete cosine transform (DCT) coefficients | COPY-MOVE + SPLICING | 96 | 99 |
| | COPY-MOVE | 97 | 99 |
| | SPLICING | 98 | 99 |
| Mantissa distribution | COPY-MOVE + SPLICING | 96.22 | 93.65 |
| | COPY-MOVE | 95.91 | 98.59 |
| | SPLICING | 97.43 | 99.13 |

*Fig. 13b*

In the case of CASIA2.0, the DCT coefficient technique's average accuracy is higher i.e. 99% than Mantissa distribution accuracy. DCT coefficient technique with SVM has given an average accuracy of 99% in CASIA 2.0.

## VI. Conclusion

The SVM algorithm for image classification achieved an impressive average accuracy of 97%, outperforming the random forest classifier across both datasets. The optimization techniques employed, including Principal Component Analysis (PCA), K-fold cross-validation, and Bayesian optimization, significantly contributed to enhancing the accuracy and overall performance of the SVM classification model, showcasing the importance of thorough optimization in machine learning applications. A comparative analysis between different feature extraction techniques revealed that, in the case of CASIA 1.0, the DCT coefficient technique outperformed the Mantissa distribution technique, achieving an average accuracy of 97% compared to 96%. Similarly, for CASIA 2.0, the DCT coefficient technique demonstrated superior accuracy, reaching 99% compared to 97% with the Mantissa distribution technique. It's worth noting that the computational complexity of the Discrete Cosine Transform (DCT) feature extraction process, particularly for large datasets, necessitated the use of GPU for efficient and accelerated computations. In conclusion, the DCT coefficients technique for feature extraction is proving to be more effective in the datasets analyzed. The efficacy of SVMs in image forgery classification with optimization techniques plays a crucial role in fine-tuning the model for accurate forgery detection.

## VII. Future Work

The forthcoming evolution in image forgery detection is on the brink of transformative progress on multiple fronts. Initially, the incorporation of sophisticated deep learning structures, including convolutional neural networks (CNNs) and recurrent neural networks (RNNs), shows potential for amplifying the automatic learning and adaptability of detection systems to intricate patterns of image manipulation. Addressing the dynamic threat landscape, research on adversarial attacks and defences is imperative to develop techniques that evade current detection methods, necessitating the exploration of robust countermeasures and adversarial training. Embracing a holistic perspective, multimodal approaches integrating text and metadata analysis with image content present an avenue to deepen the understanding of contextual information, thereby enhancing the accuracy and reliability of forgery detection systems. Ensuring the transparency and trustworthiness of detection models calls for advancements in interpretability and prompts future work to focus on methods that elucidate the rationale behind a model's classification. The continuous evolution of large-scale datasets that mirror real-world scenarios and diverse manipulation techniques is essential for improving the generalizability of detection models, emphasizing the need for sustained efforts in dataset curation.

Moreover, the exploration of hardware-accelerated implementations and optimizations for real-time forgery detection is crucial, particularly in applications where timely detection is critical. The development of collaborative frameworks and standardized evaluation metrics stands as a key initiative to facilitate meaningful comparisons between different forgery detection approaches, fostering a cohesive and advancing research community. As image manipulation techniques evolve, future research endeavours should keep pace by exploring novel methods and corresponding detection strategies, including those introduced by emerging technologies like deepfakes. The design of user-friendly forgery detection tools accessible to non-experts, coupled with educational components to raise awareness about the prevalence of manipulated media, can contribute to a more informed and vigilant public. Finally, addressing ethical implications and legal considerations surrounding forgery detection technologies is paramount for responsible development and implementation, urging researchers to delve into the multifaceted ethical and legal dimensions of this evolving field. Collectively, these future directions underscore the dynamic and multifaceted nature of image forgery detection, charting a course toward more reliable, adaptable, and ethically sound solutions.

## REFERENCES


[1] Shi, Y.Q., Chen, C.: A natural image model approach to splicing detection. In: Proceedings of the 9th workshop on Multimedia and Security, Dallas, TX, pp. 51– 62 (2007) J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68– 73.

[2] Ng, T.-T., Chang, S.-F.: A Data Set of Authentic and Spliced Image Blocks. ADVENT Technical Report, #2032004-3, Columbia University (2004)

[3] Zhen, Z., Jiquan, K.: An effective algorithm of image splicing detection. In: Proceedings of the International Conference on Computer Science and Software Engineering, Wuhan, Hubei, pp. 1035–1039 (2008).

[4] c. li, Q. Ma, L. Xiao, M. Li, A. Zhang, Image splicing detection based on markov features in qdct domain, Neurocomputing 228. doi:10.1016/j.neucom.2016.04.068.

[5] Wei, W., Jing, D.: Image tampering detection based on stationary distribution of Markov chain. In: 17th IEEE International Conference Image Processing (ICIP 2010), Hong Kong, pp. 2101–2104 (2010).

[6] A. J. Fridrich, B. D. Soukal, A. J. Lukáš, Detection of copy-move forgery in digital images, in : in Proceedings of Digital Forensic Research Workshop, 2003

[7] C. S. Prakash, A. Kumar, S. Maheshkar, V. Maheshkar, An integrated method of copy-move and splicing for image forgery detection, Multimedia Tools Appl. 77 (20) (2018) 26939–26963. doi:10.1007/s11042-018-5899-3.URL https://doi.org/10.1007/s11042-018-5899-3

[8] J. Dong, W. Wang and T. Tan, "CASIA Image Tampering Detection Evaluation Database," 2013 IEEE China Summit and International Conference on Signal and Information Processing, Beijing, China, 2013, pp. 422-426, doi: 10.1109/ChinaSIP.2013.6625374.