Image Forgery Detection of Spliced Image Class in Instant Messaging Applications

Michelle Theresia
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
michelle.theresia17@gmail.com

Rinaldi Munir
School of Electrical Engineering and Informatics
Institut Teknologi Bandung
Bandung, Indonesia
rinaldi@informatika.org

***Abstract*— As of 2021, smartphone user reached 79.84% of the world's population and 2.52 billion of them are active using instant messaging applications. As a result, the production and distribution of digital data exploded, and digital images were no exception. It also encourages the development of image manipulation techniques. On the other hand, image manipulation technology is also used to falsify information. This action is known as digital image forgery. There are several digital image forgery techniques and one of the most popular is image splicing. There are several methods for detecting spliced images and they are divided into traditional and deep learning. In recent publishing, proposed methods are mostly deep learning-based as they are able to learn features more generally. In this research, a modification is done on deep learning-based method for image splicing detection proposed by Meena & Tyagi (2021) in order for the method to detect compressed images from instant messaging applications by training compressed datasets through the Whatsapp application. The method consists of 3 stages, which are extraction of the input image's noise residual using noiseprint model, feature extraction using ResNet-50, and classification, which is then applied to a desktop application. The solution is implemented using the Python programming language with some libraries: Tensorflow for noiseprint models, Keras for ResNet-50, PyCaret for classification, and TKinter for interfaces. The experiment done is to determine the classification model that has the highest accuracy using PyCaret library. From the result, it was found that the classification model with the highest level of accuracy is the Random Forest Classifier, which is 85.19%. However, the validation of the modified image splicing detection method using 100 DSO-1 datasets compressed via the WhatsApp application was unsuccessful because the accuracy was below the success criteria. On the other hand, the desktop application functionality is fulfilled and running well.**

Keywords*— *instant messaging application; digital image forgery; image splicing; deep learning; compression.

I. INTRODUCTION

By 2021, smartphone users reached 79,84% world population and 2,52 billion among them are using instant messaging applications actively. Hence, the production and distribution of digital data exploded, including digital image. It also promotes the development of image manipulation technology and tools, such as Adobe Photoshop, GIMP, etc., enabling people to edit photos with ease. The advancement of digital image manipulation techniques has positive and negative impacts. On one hand, it facilitates the beautification of photos and thereby encourages people to express and share their ideas on visual arts of photo editing; on the other hand, it is much easier to forge the content of a given image without leaving any visible clues and thus helps forgers to deliver fake information [1]. The action of using image manipulation to deliver fake information is also known as image forgery.

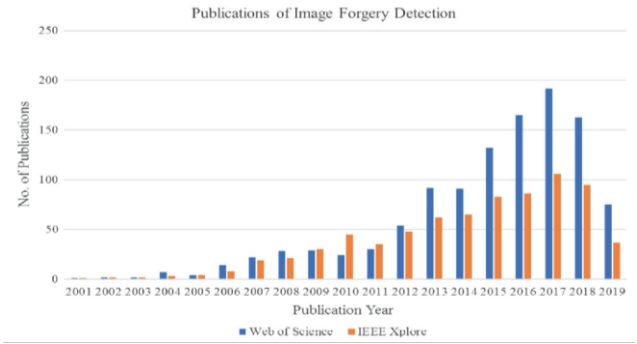

Fig 1. Number of publications per year in digital image forensics over past 19 years in IEEE (ieeexplore.org) and Elsevier (Sciencedirect.com) libraries [5]

There are 4 types of digital image forgery, which are image splicing, copy-move, morphing, and retouching [2]. Image splicing and copy-move forgery are the two most common ways to tamper with the images [3]. Image splicing manipulates images by copying a region from one image (i.e., the donor image) and pasting it onto another image (i.e., the host image) [4]. Therefore, research in digital image forensics field has increased.

The number of research publications in the digital image forensics from two different libraries, IEEE and Elsevier, has specifically increased from 2000 to 2019 as shown in Fig. 1. The method in digital image forensics is also known as digital image forgery detection. Digital image manipulation detection techniques can be divided into two methods, which are active and passive approaches. In the active approach, certain information is embedded inside an image during the creation in form of digital watermark, hence the drawback of this approach

is that a watermark must be inserted at the time of recording, which would limit to specially equip digital cameras [6]. In the passive approach, there is no pre-embedded information inside an image during the creation, but works purely by analyzing the binary information of an image [6]. Nowadays, most of the manipulated images found are obtained from different platforms. So, it is mostly impossible to find the origin of a forged image. Hence, active forgery detection methods are less useful, or in other words, passive forgery detection methods are more relevant to detect a digital image forgery, including in instant messaging applications.

Passive forgery detection techniques can be divided into two: hand-crafted and deep learning. Hand-crafted techniques rely on experts to determine extracted features from input image. Some of the popular hand-crafted techniques are, discrete wavelet transform (DWT), contourlet transform, Hilbert-Huang transform, local binary pattern (LBP), discrete cosine transform (DCT) [3]. However, recent developments of passive forgery detection mostly are based on deep learning because it uses neural network to identify more generalized features dan has better performance, although it needs larger input data. One of the most popular deep learning architectures for image classification, such in image forgery detection, is convolutional neural network (CNN).

Usually, passive forgery detection methods are specific to certain techniques, for example image splicing detection. One of the image splicing detection methods that utilizes deep learning, specifically CNN, is the noiseprint model [7]. This model overcomes the problem of the CNN denoising model (DnCNN) [8], namely the absence of image data paired with the noise by identifying the camera model to distinguish the spliced part and the original part of the image by using the knowledge that images from the same camera model have the same noise, while images from different camera models have different noise. As a result, the CNNs in noiseprint model are trained to minimalize the error distance for authentic sample and maximize the error distance for spliced sample.

Reference [3] developed noiseprint model by adding transfer learning using ResNet-50 to overcome the lack of public image splicing data and classify using SVM with RBF kernel in 2021. The method has proven to have a higher accuracy compared to other existing methods, 97.24% to be exact. However, it is not yet known whether this method can handle compression attack on spliced image which commonly found in instant messaging application. In this paper, research will be conducted to utilize the method to detect compressed spliced images in instant messaging applications.

II. RELATED WORK

A. Passive image forgery detection techniques

As mentioned before, there are two approaches in digital image forgery detection techniques: active and passive. Passive image forgery detection techniques roughly grouped into five categories [6].

The first category is the pixel-based image forgery detection. Pixel-based techniques accentuate on the pixels of the digital image and are generally classified into four sorts, such as copy-move, splicing, resampling and statistical [6].

The second category is format-based image forgery detection. Format based techniques are mainly based on image formats, in which JPEG format is preferable. Statistical correlation introduced by specific lossy compression schemes, which is helpful for image forgery detection. These techniques can be partitioned into three sorts such as JPEG quantization, Double JPEG and JPEG blocking. If the image is compressed then it is exceptionally hard to identify fraud however these techniques can detect forgery in the compressed image.

Whenever we take a picture from a digital camera, the picture moves from the camera sensor to the memory and it experiences a progression of processing steps, including quantization, color correlation, gamma correction, white adjusting, filtering, and JPEG compression. These processing steps from capturing to saving the image in the memory may shift on the premise of camera model and camera antiques. So, the third category, camera-based image forgery detection techniques work on this standard. These methods can be separated into four classes such as chromatic aberration, color filter array, camera response and sensor noise.

The fourth category is physical environment-based image forgery detection. These techniques basically based on three dimensional interactions between physical object, light and the camera. Consider the creation of a forgery showing two movie stars, rumored to be romantically involved, strolling down a nightfall shoreline. Such a picture may be made by grafting together individual pictures of each movie star. In this manner, it is frequently hard to exactly match the lighting effects under which each individual was initially captured. Contrasts in lighting across an image can be utilized as proof of altering. These techniques work on the basis of the lighting environment under which an article or picture is caught. Lighting is very important factor for capturing an image. These techniques are isolated into three classifications such as light direction (2-D), light direction (3-D) and light environment.

The last category is Geometry-based image forgery detection. These techniques basically based on principal point i.e. projection of the camera center onto the image plane, that make measurement of the object in the world and their position relative to camera. Grooves made in firearm barrels confer a twist onto the shot for increased accuracy and range. These grooves acquaint to some degree particular markings to the bullet fired, and can consequently be utilized with a particular handgun. In the same soul, several image forensic techniques have been produced that particularly display relics presented by different phases of the imaging procedure. Geometry-based image forgery detection methods are separated into two classes such as principle point and metric measurement.

B. Noiseprint model

The base architecture of noiseprint model [7] is feed forward denoising convolutional neural network (DnCNN) [8]. The architecture is shown in Fig. 2.

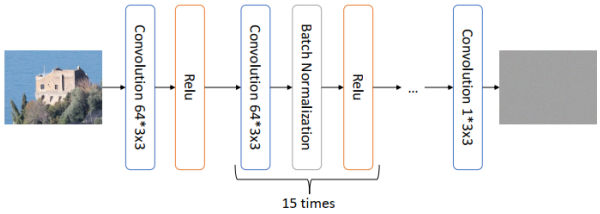

Fig 2. DnCNN architecture in Noiseprint model [7]

Image denoising is to remove noise from a noisy image, caused by the influence of environment, transmission channel, and other factors during acquisition, compression, and transmission, leading to distortion and loss of image information, so as to restore the true image [9]. DnCNN utilized convolutional neural network without pooling layer for image denoising. There are 3 layers in DnCNN with depth D as shown in Fig. 3. (i) Conv+ReLU: for the first layer, 64 filters of size $3 \times 3 \times c$ are used to generate 64 feature maps, and rectified linear units (ReLU, $max(0, \cdot)$) are then utilized for nonlinearity. Here c represents the number of image channels, i.e., $c = 1$ for gray image and $c = 3$ for color image. (*ii*) Conv+BN+ReLU: for layers $2 \sim (D - 1)$, 64 filters of size 3×3×64 are used, and batch normalization is added between convolution and ReLU. (*iii*) Conv: for the last layer, c filters of size $3 \times 3 \times 64$ are used to reconstruct the output [8].

The DnCNN is trained with a large number of paired input-output patches, where the input is a noisy image patch and the output its noise content. Noiseprint model resume the training by submitting new paired patches, where the input is a generic image patch, and output the corresponding noiseprint. The only problem is that the authors have no model of the image noiseprint therefore they cannot produce the output patches necessary for this training procedure.

The solution for the problem was relying on the information that image patches coming from the same camera model should generate similar noiseprint patches, and image patches coming from different camera models dissimilar noiseprint patches [7]. Noiseprint model leverages this knowledge by training the network to generate the desired noise residual where not only the scene content but all non-discriminative information is discarded, while discriminative features are enhanced [7]. The authors have proposed a Siamese based model, as shown in Fig. 3, formed by the parallel of two identical DnCNNs, named as residual neural network, both have the same architecture and the same weights and was trained by feeding the sequence of image patches in both the CNNs. The Siamese network minimizes the error distance for positive samples (image patches from the same camera models) and maximizes the error distance for negative samples (image patches from the different camera models).

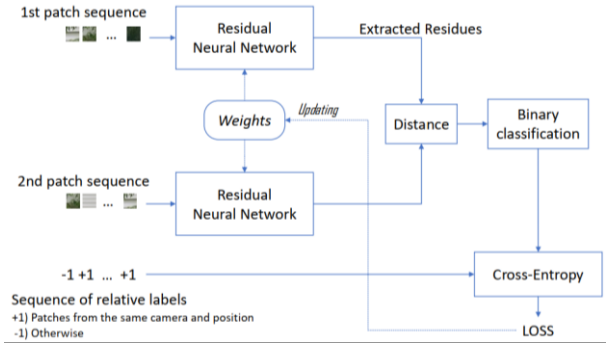

Fig 3. Siamese architecture for training noiseprint model [7]

The spliced images and their noiseprint output examples are shown in Fig. 4. Note that the top spliced image was made by inserting the aircraft object in the original image and the bottom spliced image was made by inserting the woman figure on the rightmost in the original image. It can be easily seen that the noiseprint outputs highlights the spliced part (aircraft and rightmost woman figure). Therefore, noiseprint output can be used to show localization of a spliced image.

Fig 4. Spliced image (left) with the corresponding noise residual map (right) obtained using the Noiseprint model [7]

C. Noiseprint model development with ResNet-50

Deep learning-based methods for image splicing detection require large data to train the model, meanwhile the image splicing datasets available in public domain are not so large in terms of the number of images. Reference [3] proposed a method to solve the problem by applying transfer learning for feature extraction. Transfer learning in feature extraction takes advantage of learned feature maps of a pre-trained model to extract meaningful features from new samples without having to start from scratch by training a large model on a large dataset. The pre-trained model used in the method is ResNet-50, specifically only the first 49 layers, since the last layer is a simple softmax layer with 1000 neurons. ResNet-50 was chosen for its simple yet robust architecture.

Fig. 5 shows the method architecture. Simply put, noiseprint model is used to obtain noise residual map from the input image, the features of the map are extracted using the pre-trained ResNet-50 network to learn the features that distinguished spliced and authentic images, and lastly feature

classification is done using support vector machine (SVM) with RBF kernel.

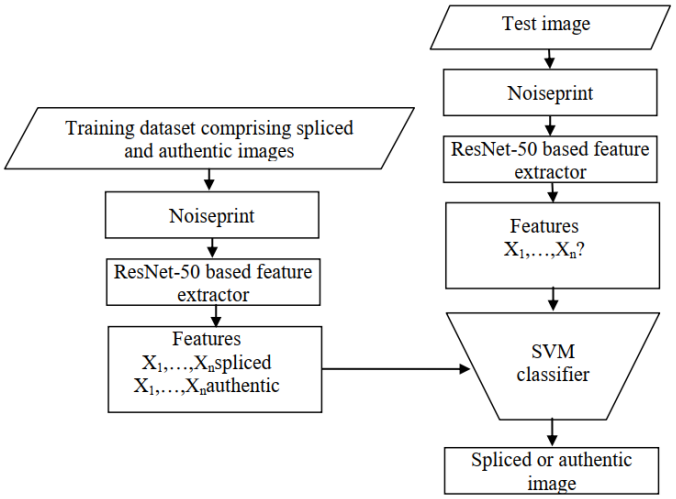

Fig 5. Deep learning-based method for image splicing detection architecture [6]

Fig. 6 shows result comparison of method [3] with existing image splicing methods using accuracy metric.

Method	Used Feature extractor	Classifier	Detection accuracy (%)
Zhao et al. [41]	Optimal chroma-like channel	SVM with Gaussian kernel	93.14
Muhammad et al. [42]	Steerable pyramid transform and LBP	SVM	96.39
Vidyadharan et al. [26]	LBP	Random forest	92.01
He et al. [12]	DCT and DWT	SVM	87.52
Alahmadi et al. [43]	DCT and LBP	SVM	96.60
Hakimi et al. [25]	LBP, wavelet transform, and PCA	SVM	95.13
Manu et al. [44]	DCT	SVM	95.98
Zhang et al. [13]	DCT and LBP	SVM	91.38
The proposed method	**ResNet-50**	**SVM with RBF kernel**	**97.24**

Fig 6. Experimental result of deep learning-based method for image splicing detection [6]

III. PROPOSED SOLUTION

A. *Modified image splicing detection method*

The method proposed by [3] was trained and tested on an uncompressed image dataset (The Columbia Uncompressed Image Splicing Dataset or CUISDE). As mentioned before, images distributed through instant messaging applications are always compressed. So, in order for the method to be able to detect compressed images, the method will be trained and tested with the same dataset used in [3] but uploaded and downloaded through WhatsApp application. The WhatsApp application performs JPEG compression on the sent images, proven by the downloaded images converted to JPEG format and the images sizes are smaller compared to the original.

The training process is done the same way as the image splicing detection method in [3], which starts with extracting the residual noise of each image that has been redownloaded through the WhatsApp application using the noiseprint model and noiseprint image in PNG format. Each of the noiseprint image features are extracted using a pre-trained ResNet-50 so that 2048 features were obtained and concatenated with 'target' column containing the image class (authentic/spliced). These features are used to train the classification model.

The classification process is done almost the same way as the training process, which also starts with extracting the residual noise of the input image and noiseprint image in PNG format. The noiseprint image features are extracted with pre-trained ResNet-50 and 2048 features were obtained. Those features will be classified by the trained classification model. If the classification result is authentic, the method will only output the original image, meanwhile if the result is spliced, the method will output the original image and its noiseprint image which shows the localization of the spliced part. Fig. 7 shows the flow of the modified image splicing detection method.

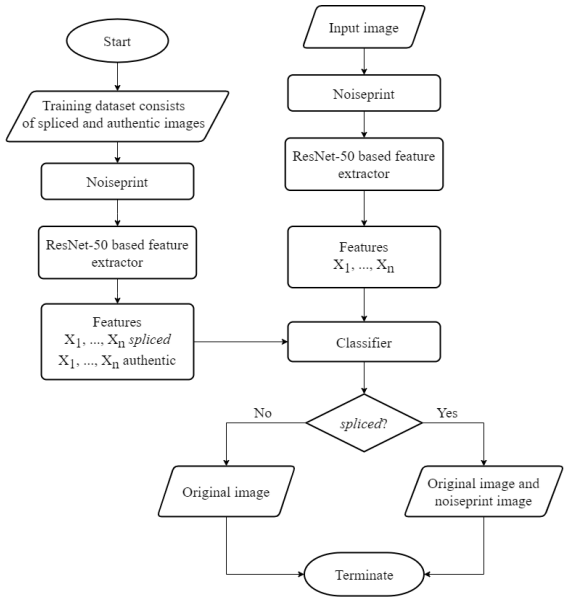

Fig 7. Proposed image splicing detection method flow

Another problem from the method in [3] is the accuracy of reproduction for this research is lower than in their experiment. Therefore, an experiment to find modifications that able to increase the accuracy in implementation is conducted. One thing that can be done is by changing the classification model.

To fulfill the requirement of image splicing detection that can be used for instant messaging applications, the proposed solution is applying the modified detection method as a desktop application.

B. *Implementation*

The proposed method and desktop application are implemented in Python language. The performance evaluation is done in Google Colaboratory. The description of training and evaluation datasets for experiments is given in the next section.

1) *Datasets and performance evaluation*

The training and testing dataset used in the experiment is the same as [3], which is The Columbia Uncompressed Image Splicing Detection Evaluation (CUISDE). The dataset consists of 2 folders: 4cam_auth for authentic images, which described as images taken with a single camera, and 4cam_splc for spliced images, which described as the combination of images from 2 different camera model. There are a total of 183 authentic and 180 spliced images stored in TIFF formats. The

pixel resolution of the images in both folders is in the range from 757×568 to 1152×768. The images are captured using four different camera models and used as the file names, namely ‘canong3’, 'nikond70', 'canonxt', and 'kodakdcs330'. Since the spliced images are the combination of 2 different camera model, the file names attached 2 camera model names. Fig. 8 shows a few examples of authentic images in the dataset and Fig. 9 shows a few examples of spliced images in the dataset.

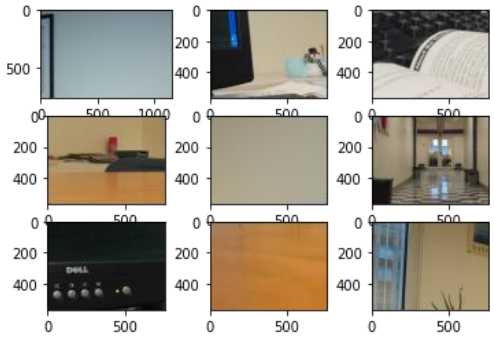

Fig 8. First nine authentic images in CUISDE dataset.

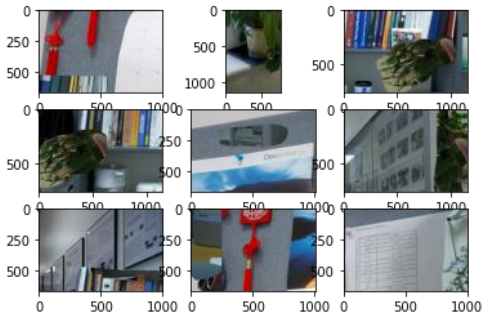

Fig 9. First nine spliced images in CUISDE dataset

All of the images are uploaded and downloaded through WhatsApp manually. However, WhatsApp only accepts JPEG/JPG, PNG, or GIF images to be sent as ‘Photos’, so the images which in TIFF formats are converted first to JPG with PIL library. Note that images converted to JPG must have compressed, but by setting quality parameter to 100 and subsampling parameter to 0, the quality of the converted images are maximum.

Since the number of the training and testing dataset is quite little, a different dataset is used for validation, which is DSO-1. DSO-1 is composed of 200 indoor and outdoor images with an image resolution of 2,048 x 1,536 pixels. Out of this set of images, 100 are original, i.e., have no adjustments whatsoever, and 100 are forged. The forgeries were created by adding one or more individuals in a source image that already contained one or more persons. The images used in validation are first 50 original and 50 spliced images. All of the images are also uploaded and downloaded through WhatsApp manually.

The evaluation metric used in the experiment is detection accuracy. It is a valid metric if the training dataset balanced in terms of each class. Since the number of authentic and spliced images in the training dataset is almost 1:1, it is valid to use detection accuracy for evaluation. The detection accuracy is defined as (1).

$$\text{Detection accuracy} = (T_p + T_n)/(T_p + T_n + F_p + F_n) \times 100 \quad (1)$$

Where T_p is the True Positive and defined as the total number of images that are correctly detected as authentic. T_n is the True Negative and defined as the total number of images that are correctly detected as spliced. F_p is the False Positive and defined as the total number of spliced images that are erroneously detected as authentic. F_n is the False Negative and defined as the total number of authentic images that have erroneously identified as spliced.

2) Image splicing detection method

The noise residual extraction of each training data uses function from Noiseprint model implementation by Paolus [10]. The code was the modification of [7] which uses TensorFlow 1 library to TensorFlow 2, so it becomes compatible with Python 3.8 above. The function used from [10] is noiseprint_blind which returns a variable to generate a noise residual map. The residual noise map is generated using the genMappUint8 function from [10] with the output parameter of the noiseprint_blind function. The map is still an array, so it saved in PNG format using the save function in the PIL library.

ResNet-50 is implemented using Keras library. Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. The function used is ResNet50. Since it is used as feature extraction there are some settings done in the parameters, such as include_top set as False, weights set as ‘imagenet’ for using pretrained ResNet-50, and pooling set as ‘max’.

The classification model is implemented using PyCaret library. The classification model used is the one that has the highest accuracy based on the result in evaluation.

3) Desktop application

The implementation for the GUI uses Tkinter library since it is a desktop application. The tkinter package (“Tk interface”) is the standard Python interface to the Tcl/Tk GUI toolkit.

IV. EVALUATION

A. Highest classification model accuracy

The purpose of this experiment is to find out which classification model has the highest accuracy.

1) Scenario

Using SVM with RBF kernel as in [3], the accuracy only reached 78,13%. So, to find the classification model with higher accuracy, PyCaret library is utilized by using compare_models function.

2) Result

Table I shows the comparison of every classification model available in the library sorted with highest accuracy.

TABLE I. CLASSIFICATION MODEL ACCURACY RESULT

	Model	Accuracy
rf	Random Forest Classifier	0.8519
et	Extra Trees Classifier	0.8395
lda	Linear Discriminant Analysis	0.8339
lr	Logistic Regression	0.8305
nb	Naive Bayes	0.8274
svm	SVM - Linear Kernel	0.8242
ridge	Ridge Classifier	0.8184
gbc	Gradient Boosting Classifier	0.8156
lightgbm	Light Gradient Boosting Machine	0.8062
knn	K Neighbors Classifier	0.7849
ada	Ada Boost Classifier	0.7697
dt	Decision Tree Classifier	0.7362
qda	Quadratic Discriminant Analysis	0.5737
dummy	Dummy Classifier	0.5091

It can be seen that the classification model with highest accuracy is Random Forest Classifier (RF), specifically 85.19%. The major difference of RF and SVM is how it determines the class of an input data. RF uses the probability of a data feature in each class, while SVM calculates the distance of the data feature to each class. However, SVM has a weakness if the number of features in each data sample exceeds the number of training data samples because it may cause overfitting. Hence, the number of features per data sample in the test is 2048, that is much more than the number of training samples, which is 326.

Another reason SVM has lower accuracy is its difficulty in determining support vector if the training data feature values are close for each class and very likely for misclassification to happen. Meanwhile, compressed images cause noise in spliced part of an image similar or in other words, the feature values of authentic and spliced images become close as well. Perhaps, this causes SVM difficult in dividing features into authentic and spliced classes. RF is superior in handling larger features since it uses probability in the classification, which causes the chance of overfitting lower than SVM.

B. Compressed images validation

The purpose of this experiment is to validate the accuracy of the finalized method to unseen samples.

1) Scenario

As mentioned in datasets, the validation dataset used is 100 images from DSO-1 dataset. The images are uploaded and downloaded through WhatsApp manually, so they are compressed. Same as training process, the images noiseprint are extracted with noiseprint model and each noiseprint image features are extracted with pre-trained ResNet-50, lastly RF will classify each which result will be concatenated to the features and target column as Label and Score. The metric evaluation done by comparing the Label and target column.

2) Result

The detection accuracy of RF model is 0.57, or 57%. Fig. 10 shows the confusion matrix of classification result. It can be seen that 38 out of 50 authentic images are correctly classified, or 76%. Meanwhile, only 19 out of 50 spliced images are correctly classified or 38%.

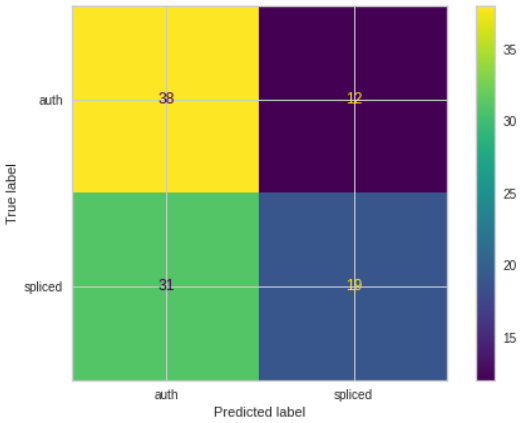

Fig 10. Confusion matrix of RF classification result

The possible cause of the low accuracy is that the WhatsApp application applies JPEG compression, proven by all image files in JPG or PNG format are downloaded as JPEG format. JPEG compression is a lossy compression method, especially in high-frequency parts, so it is not surprising that JPEG compression greatly affects image noise [11]. Moreover, it is possible that JPEG compression reduces discriminatory anomalies between block pairs, removing camera fingerprints for a CNN patch-based approach [12]. Unfortunately, noiseprint model [7] approach is to highlight the CNN patch-based camera fingerprint. So, it can be said that the modification of the image splicing detection method by Meena & Tyagi (2021) is not robust enough to handle compression on spliced images.

V. CONCLUSION

In this paper, a modification on a deep learning-based method to detect the image splicing forgery is proposed to handle compression attack on instant messaging applications. Noiseprint model is one of camera-based image forgery detection techniques. Unfortunately, even by training the model with compressed images and improve its accuracy by changing the classification model, the method is not robust enough to detect compression attack. The main reason is JPEG compression on instant messaging applications reduces discriminatory anomalies between block pairs, removing camera fingerprints, while noiseprint model approach to highlight the CNN patch-based camera fingerprint. On the other side, it is very feasible to apply the method as desktop application. So, it is worth to do an experiment to use other approach, such as format-based image forgery detection for future work and it will be better if the application are accessible on multiple platform since instant messaging user are using mobile platform, thus they don't need to switch platforms.

ACKNOWLEDGMENT

The author would like to thank God the Almighty for His blessings throughout this research work so it can be completed. The author would also like to thank Dr. Ir. Rinaldi, M.T. for his valuable and constructive guidance and teaching during the whole process of this research. The author also express

gratitude to family and friends for their help and support in the making of this paper.

REFERENCES

[1] L. Zheng, Y. Zhang and V. Thing, "A survey on image tampering and its detection in real-world photos", Journal of Visual Communication and Image Representation, vol. 58, pp. 380-399, 2019. Available: 10.1016/j.jvcir.2018.12.022 [Accessed 23 June 2022].

[2] A. Abidin, H. Majid, A. Samah and H. Hashim, "Copy-Move Image Forgery Detection Using Deep Learning Methods: A Review", 2019 6th International Conference on Research and Innovation in Information Systems (ICRIIS), 2019. Available: 10.1109/icriis48246.2019.9073569 [Accessed 23 June 2022].

[3] K. Meena and V. Tyagi, "A Deep Learning based Method for Image Splicing Detection", Journal of Physics: Conference Series, vol. 1714, no. 1, p. 012038, 2021. Available: 10.1088/1742-6596/1714/1/012038.

[4] R. Salloum, Y. Ren and C. Jay Kuo, "Image Splicing Localization using a Multi-task Fully Convolutional Network (MFCN)", Journal of Visual Communication and Image Representation, vol. 51, pp. 201-209, 2018. Available: 10.1016/j.jvcir.2018.01.010.

[5] M. Ahmad and F. Khursheed, "Digital Image Forgery Detection Approaches: A Review", Algorithms for Intelligent Systems, pp. 863-882, 2021. Available: 10.1007/978-981-33-4604-8_70 [Accessed 23 June 2022].

[6] A. Kashyap, R. Parmar, M. Agarwal and H. Gupta, "An evaluation of digital image forgery detection approaches", arXiv preprint, 2017. Available: https://arxiv.org/abs/1703.09968. [Accessed 23 June 2022].

[7] D. Cozzolino and L. Verdoliva, "Noiseprint: A CNN-Based Camera Model Fingerprint", IEEE Transactions on Information Forensics and Security, vol. 15, pp. 144-159, 2020. Available: 10.1109/tifs.2019.2916364.

[8] K. Zhang, W. Zuo, Y. Chen, D. Meng and L. Zhang, "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising", IEEE Transactions on Image Processing, vol. 26, no. 7, pp. 3142-3155, 2017. Available: 10.1109/tip.2017.2662206.

[9] L. Fan, F. Zhang, H. Fan and C. Zhang, "Brief review of image denoising techniques", Visual Computing for Industry, Biomedicine, and Art, vol. 2, no. 1, 2019. Available: 10.1186/s42492-019-0016-7

[10] Paolus (2021) Noiseprint: a CNN-based camera model fingerprint [Source code]. https://github.com/Paoulus/noiseprint

[11] T. Julliand, V. Nozick and H. Talbot, "Image Noise and Digital Image Forensics", Digital-Forensics and Watermarking, pp. 3-17, 2016. Available: 10.1007/978-3-319-31960-5_1 [Accessed 25 June 2022].

[12] B. Diallo, T. Urruty, P. Bourdon and C. Fernandez-Maloigne, "Robust forgery detection for compressed images using CNN supervision", Forensic Science International: Reports, vol. 2, p. 100112, 2020. Available: 10.1016/j.fsir.2020.100112.