International Journal of Scientific Research in Engineering & Technology
Volume6, Issue6 (November-December 2026), PP: 62-68.
https://www.doi.org/10.59256/ijsreat.20260606010
www.ijsreat.com
ISSN No: 2583-1240

AI-Based Image Forgery Detection System Using EfficientNet-B0 and Grad-CAM Explanations with a Web-Based Interface

Bhoomika J[1], Yashaswini Y V[2], Lakheesh L Lakkaiah[3], Vani Patil [4], Puneeth S P[5]
[1,2,3,4]*Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, Karnataka, India.*
[5]*Professor, Project Mentor, Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, Karnataka, India.*

To Cite this Article: *Bhoomika J[1], Yashaswini Y V[2], Lakheesh L Lakkaiah[3], Vani Patil [4], Puneeth S P[5], "AI-Based Image Forgery Detection System Using EfficientNet-B0 and Grad-CAM Explanations with a Web-Based Interface", International Journal of Scientific Research in Engineering & Technology, Volume 06, Issue 06, November-December 2026, PP:62-68.*

Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: *Digital photographs are now routinely used as evidence in news reporting, courts, insurance claims and identity checks, yet inexpensive editing software and generative models make it easy to alter them without visible traces. Manual inspection by experts does not scale, and many classical detectors fail once an image has been recompressed or resized. This paper presents the design and prototype implementation of an AI-based image forgery detection system that labels an uploaded picture as genuine or forged, reports a confidence value, and shows a Grad-CAM heat map of the regions that most influenced the decision. The system couples a Streamlit front end with a Flask back-end service. Uploaded images are resized to 224×224 pixels, smoothed with a Gaussian filter ($\sigma = 0.5$), normalised in the manner expected by EfficientNet, and passed to an EfficientNet-B0 classifier. A history page and a downloadable PDF report are provided for registered users. The prototype was exercised on sample inputs, and the captured screens show end-to-end operation, including a low-margin decision (56.43 % confidence) whose Grad-CAM map illustrates why a confidence score alone is not sufficient for forensic use. A quantitative evaluation protocol on public tampering benchmarks is specified as the next phase of the work.*

Keywords: *image forgery detection, convolutional neural network, EfficientNet-B0, Grad-CAM, explainable AI, copy-move, splicing, Streamlit, Flask.*

I. INTRODUCTION

Photographs have long been treated as a faithful record of events, and that trust is now under strain. A few minutes with a mainstream editor, or a single prompt to a generative model, is enough to remove a person from a scene, insert an object that was never there, or fabricate a face. The consequences reach well beyond entertainment: altered images circulate as news, are submitted as claims, and are used to impersonate people during remote identity checks. Reviews of the field describe image and video forensics as a race in which the means of manipulation improve at least as quickly as the means of detection [1, 2].

Early forensic work did not try to learn the difference between real and edited pictures. It looked instead for physical or statistical inconsistencies: duplicated blocks left behind by cloning [3], disturbed sensor-noise statistics, or traces of repeated JPEG compression [1]. These cues are informative but fragile, because a simple resize or a re-save at a different quality can erase them. Deep learning changed the picture by allowing a network to *learn* manipulation traces directly from examples [4, 5, 6], and this is the approach taken in the project reported here.

The system described in this paper is a student-built, end-to-end prototype rather than a new detection algorithm. Its aim is to show that a compact, widely available backbone can be wrapped in a usable application in which the user uploads a picture, receives a verdict with a confidence value, and is shown *where* in the picture the network looked. The contributions are as follows.

- A modular architecture that separates the user interface, the inference service, the model, and the storage layer, so each can be replaced independently (Section 4).
- A documented preprocessing chain and an EfficientNet-B0 based binary classifier, with Grad-CAM used to make each decision inspectable (Sections 4.3–4.5).
- A working web application with registration, login, upload, result, history and PDF-report functions, illustrated with screenshots of the running system (Section 5).
- An honest account of what the prototype does and does not yet demonstrate, together with the evaluation protocol that will be used to measure it on public benchmarks (Sections 6–7).

The rest of the paper is organised as follows. Section 2 surveys related work. Section 3 lists the requirements. Section 4 presents the design, and Section 5 the implementation and user interface. Sections 6 and 7 discuss observed behaviour and the planned evaluation. Sections 8–10 close with applications, limitations and conclusions.

II. RELATED WORK

2.1 Kinds of image forgery

Three manipulation families recur throughout the literature and in the scope of this project. *Copy-move* forgery duplicates a region of an image and pastes it elsewhere in the same image, typically to conceal an object or multiply a crowd. *Splicing* composes content from two or more different photographs. *Synthetic or generated* content, including face swaps and entirely generated faces, is produced by neural networks rather than by cutting and pasting [7]. A forensic system may be asked to decide only whether an image has been altered, or additionally to say where.

2.2 Hand-crafted forensic methods

Surveys of passive forensics group classical methods by the trace they exploit [1]. For copy-move detection, Fridrich et al. proposed sliding a block over the image, describing each block by quantised DCT coefficients, and sorting the descriptors lexicographically so that duplicated blocks become neighbours [3]. For splicing, Cozzolino et al. described Splicebuster, which characterises the local noise residual with co-occurrence statistics and flags regions that deviate from the dominant statistics of the image, without needing examples of tampered images for training [8]. These methods are interpretable and light-weight, but each targets a specific kind of trace and degrades when the trace is disturbed by post-processing.

2.3 CNN-based manipulation detection

Bayar and Stamm introduced a constrained first convolutional layer whose filters are forced to behave as prediction-error filters, which suppresses image content and lets the following layers concentrate on manipulation artefacts [4, 9]. Zhou et al. combined an RGB stream with a noise-residual stream in a two-stream Faster R-CNN, allowing the detector to use both visible tampering evidence and noise inconsistencies and to localise the altered region with a bounding box [5]. ManTra-Net treats a manipulation as an anomaly with respect to the local features of the same image and produces a pixel-level mask for many manipulation types [10]. A recent survey organises the deep-learning literature by network design, supervision level and benchmark, and discusses open challenges, including generalisation to manipulations unseen during training [6].

2.4 Face manipulation and deepfakes

Generated and swapped faces have produced their own sub-field. Tolosana et al. review manipulation categories such as identity swap, expression swap, attribute editing and entire face synthesis, together with detection methods and databases [7]. Rössler et al. released a large benchmark of videos manipulated by several methods and showed that detectors degrade at low video quality [11]. Caldelli et al. fed optical-flow fields to a CNN and examined behaviour on manipulations not seen during training [12]. Since the present system receives single still images, deepfake faces are treated here as one category of manipulation to be addressed once suitable training data are added.

2.5 Backbone and explanation techniques

Efficient Net scales network depth, width and input resolution together using one compound coefficient, and its baseline B0 model reaches 77.1 % top-1 accuracy on ImageNet with about 5.3 million parameters [13], which makes it attractive for deployment on ordinary hardware. Grad-CAM produces a coarse localisation map for a chosen class from the gradients flowing into the last convolutional feature maps, without changing the network [14]. Its use in a forgery detector is natural, because a verdict that cannot be inspected is hard to defend.

2.6 Datasets

Training and evaluation of image-tampering detectors rely on collections of authentic images paired with tampered versions. The CASIA tampering database [15] is among the most widely used for splicing and copy-move evaluation. Backbones for such detectors are commonly initialised from models pretrained on ImageNet [16].

2.7 Gap addressed

Reading across these sources, three practical gaps motivate this project: (i) many research detectors are delivered as scripts, not as applications that a non-specialist can operate; (ii) outputs are often a bare label, leaving the user unable to judge how much weight to give it; and (iii) heavier architectures are costly to deploy. The prototype addresses these by combining a compact backbone, a visual explanation, and a conventional web front end.

III. SYSTEM REQUIREMENTS

3.1 Functional requirements

The project documentation defines eight functional groups: (1) user registration, login and password reset; (2) image upload with format checking for JPG, JPEG and PNG; (3) image processing, comprising preprocessing and feature extraction; (4) forgery detection, which classifies the image and targets copy-move, splicing and deepfake manipulation; (5) database management for user records and analysis history; (6) visualisation and reporting, including highlighted regions and a downloadable report; (7) administrator control over users, uploads and permissions; and (8) security features covering access restriction, data protection and activity logging.

3.2 Non-functional requirements

The system is required to respond quickly enough for interactive use, behave consistently across varied inputs, scale to more users and uploads, present a simple interface, protect user data and uploaded images, tolerate noise and compression, and remain easy to maintain and extend.

3.3 Hardware and software

Table 1 summarises the platform specified for development and deployment.

Item	Specification / purpose
Computer	Intel Core i5 or AMD Ryzen 5 (or better); 8 GB RAM minimum (16 GB recommended); 256 GB HDD/SSD or larger; optional NVIDIA GPU with at least 4 GB VRAM
Operating system	Windows 10/11 (macOS also listed in the synopsis)
Language and ML	Python 3.8+; TensorFlow/Keras (PyTorch also listed) for the model [17]
Image and data	OpenCV and PIL for images; NumPy, pandas and scikit-learn for data handling
Application layer	Streamlit (front end), Flask (back end), SQLite or MySQL (database); VS Code or PyCharm for development

Table 1. Development platform specified for the prototype.

IV. SYSTEM DESIGN

4.1 Architecture

The architecture is a pipeline with a persistent store (Fig. 1). A user interacts only with the Streamlit front end, which sends the uploaded image to the Flask service. The service preprocesses the image, runs the classifier, computes the Grad-CAM map, and returns the verdict, the confidence value and the overlay. Results are written to the database, from which the history page and the PDF report are generated. The front end also shows an "API Online" indicator, so a user can see whether the inference service is reachable before submitting an image.

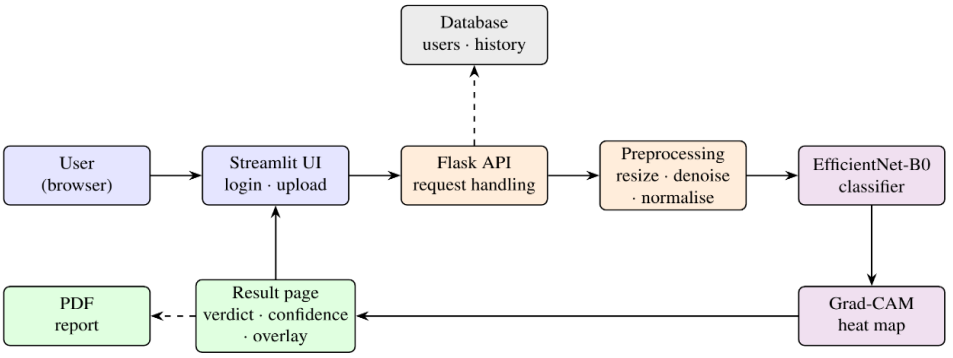

Fig. 1. Architecture of the AI-based image forgery detection system. Solid arrows show the processing path; dashed arrows show storage and report generation.

4.2 Modules

User and input module. The user logs in, selects an image, and previews it. Accepted formats are JPG, JPEG and PNG; the interface displays the file name, size and the dimensions after resizing, which allows the user to confirm what was actually analysed.
Preprocessing module. Applies the steps detailed in Section 4.3, shown to the user as a checklist beside the preview.
Classification module. Runs the EfficientNet-B0 model and returns a binary decision and a confidence value (Section 4.4).
Explanation module. Computes a Grad-CAM map and blends it with the original image (Section 4.5).
Storage and reporting module. Stores accounts and analysis records, supplies the history page, and creates a downloadable PDF report of a given result.
Administration and security. Role-based accounts distinguish ordinary users from administrators; the interface displays the signed-in user name and role. Access to analysis pages requires login.

4.3 Preprocessing

The interface lists four preprocessing steps for every upload. Let I be the uploaded image.
Resizing. The image is resampled to the network input size, $I_{224} = R(I)$. A fixed size lets one trained model serve inputs of any resolution, at the price that fine high-frequency traces may be attenuated by down-sampling.
Gaussian denoising. A Gaussian low-pass filter with $\sigma = 0.5$ is applied,

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), \qquad I_d = G_\sigma * I_{224}. \tag{1}$$

With such a small σ the kernel is nearly an identity operator and removes only isolated pixel-level noise; this is a

deliberate design choice, since heavy smoothing would erase the very residual patterns on which forensic networks rely [9].

Efficient Net normalisation. Pixel values are mapped to the range that the EfficientNet implementation expects, $\hat{I} = N(I_d)$.

Batch dimension. The tensor is expanded to $X \in \mathbb{R}^{1\times224\times224\times3}$ so that it can be fed to the network as a batch of one.

4.4 Classification model

The classifier is built on EfficientNet-B0 [13]. The baseline network begins with a stem convolution, continues through seven stages of mobile inverted-bottleneck (MBConv) blocks with squeeze-and-excitation, and ends in a 1×1 convolution and global average pooling. EfficientNet derives larger variants by scaling depth $d = \alpha^{\varphi}$, width $w = \beta^{\varphi}$ and resolution $r = \gamma^{\varphi}$ with one coefficient φ under the constraint $\alpha\beta^2\gamma^2 \approx 2$ [13]; B0 is the unscaled baseline, with a native input of 224×224, which matches the resolution used in the preprocessing chain above. The pooled features feed a binary output stage that produces the probability of each class (Fig. 2). The reported confidence is the percentage assigned to the predicted class, and the image is labelled forged when the forged-class probability exceeds a decision threshold τ, conventionally 0.5:

$$\hat{y} = \begin{cases} \text{Forged}, & p_{\text{forged}} \geq \tau, \\ \text{Genuine}, & \text{otherwise}. \end{cases} \tag{2}$$

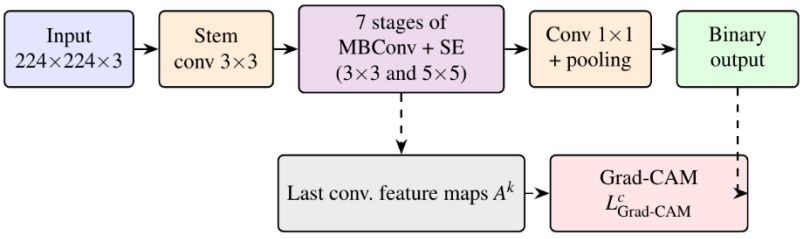

Fig. 2. Inference path. The classifier is an EfficientNet-B0 backbone with a binary output; Grad-CAM (dashed) reuses the last convolutional feature maps and the gradient of the class score.

4.5 Grad-CAM explanation

For a class c with score y^c and last-layer feature maps A^k, Grad-CAM weights each map by the spatially averaged gradient of the score [14],

$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A_{ij}^k}, \qquad L^c_{\text{Grad-CAM}} = \text{ReLU}\Big(\sum_k \alpha_k^c A^k\Big), \tag{3}$$

where Z is the number of spatial positions. The map is up-sampled to the image size, colour-coded from blue (low relevance) to red (high relevance), and blended with the original. The interface displays the original and the overlay side by side, captioned "Heatmap – red = suspicious regions". It is important to read this correctly: the map shows where the network found evidence for its decision, which is not the same as a verified segmentation of the tampered area. Treating it as ground truth would overstate what the method guarantees.

V.IMPLEMENTATION AND USER INTERFACE

The application is written in Python. The front end uses Streamlit with a dark theme and a sidebar for navigation, and the back end is a Flask service that hosts the model. The sidebar carries the application name together with the label "EfficientNetB0 · GradCAM", the API status, and the navigation entries.

Authentication. Unauthenticated visitors see only the login and registration pages (Fig. 3). Registered users sign in with e-mail and password; the password field can be revealed or masked.

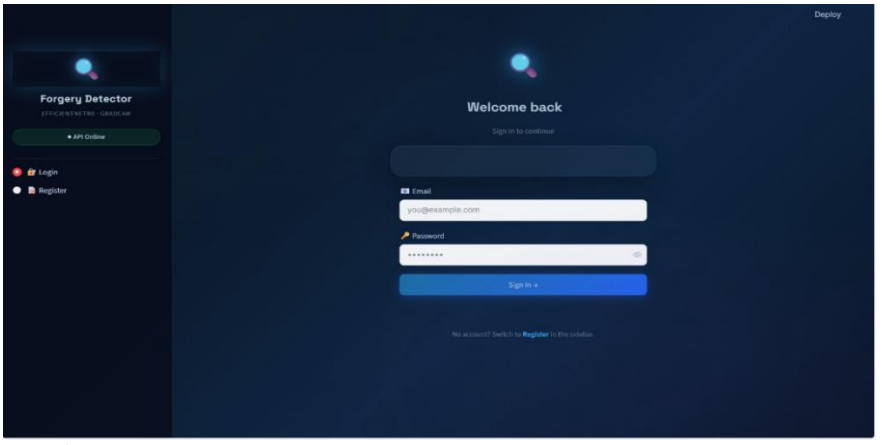

Fig. 3. Login page of the prototype, with the sidebar showing the model label, API status and the Login/Register switch.

Upload and detect. After sign-in, the sidebar shows the user name and role and offers Home, Upload & Detect, Result and History pages, and a Logout button. On the Upload & Detect page the user selects an image, sees a preview with file name, size and dimensions, reviews the preprocessing checklist, and presses *Run Detection* (Fig. 4).

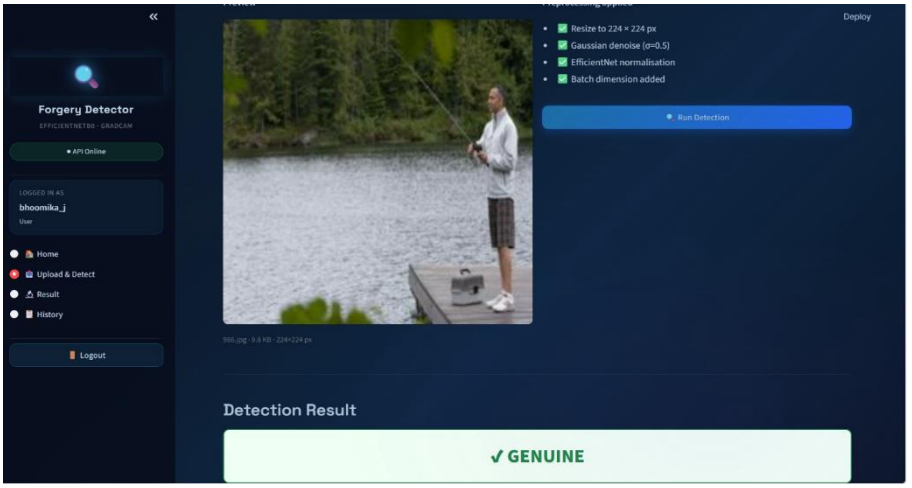

Fig. 4. Upload & Detect page: preview, preprocessing checklist, Run Detection button and a "Genuine" verdict banner for a 224×224 JPEG sample.

Result presentation. The verdict is shown as a coloured banner (green "GENUINE" or red "FORGED"), followed by a confidence percentage and a progress bar (Fig. 5). Below it, the Grad-CAM panel presents the original and the heat map next to each other, and a button downloads a PDF report of the analysis (Fig. 6). A History page in the sidebar is provided for reviewing earlier analyses.

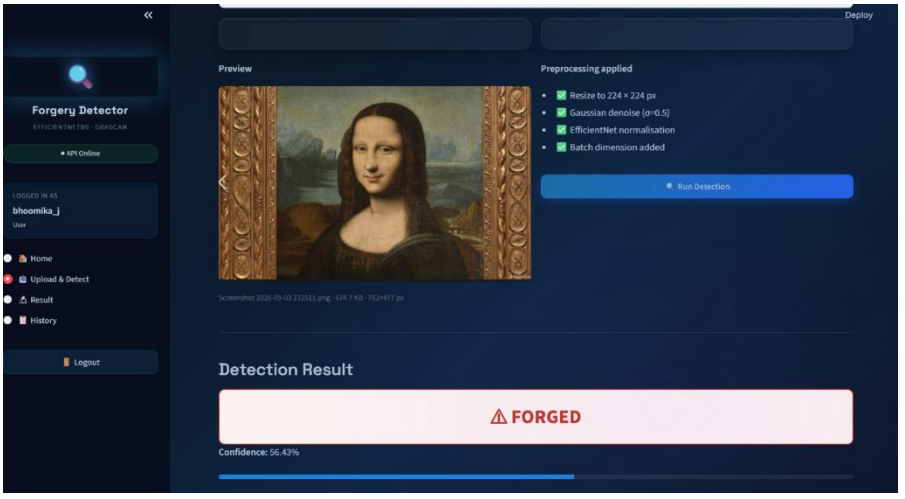

Fig. 5. A "Forged" verdict with a confidence of 56.43 %, produced for a 762×477 pixel image of a painted artwork.

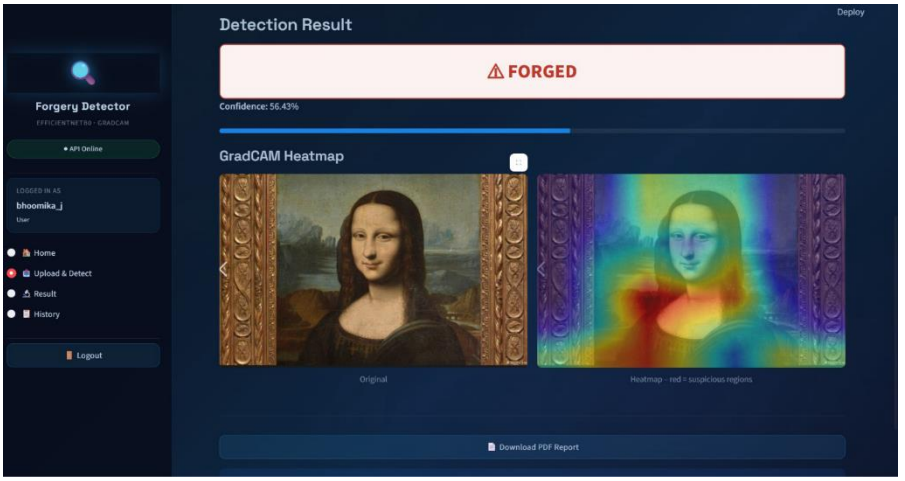

Fig. 6. Result page with the Grad-CAM panel (original on the left, heat map on the right) and the PDF-report download option.

VI. OBSERVED BEHAVIOUR AND DISCUSSION

At the time of writing, the project is at the end of its first phase and the available evidence consists of recorded sessions of the running prototype, not of a benchmark run. Table 2 lists what those sessions show. The figures in this paper are therefore evidence that the pipeline operates end to end; they are not accuracy measurements, and no accuracy, precision or recall value is claimed.

Input	Size	Decision	Confidence	Heat map
Photograph (986.jpg)	9.8 KB, 224×224	Genuine	not visible in the captured screen	not captured
Artwork image (screenshot)	674.7 KB, 762×477	Forged	56.43 %	strongest over the lower centre of the figure, weakest on the frame

Table 2. Behaviour of the prototype on the two recorded sessions.

Three observations follow. First, the interface completes the full chain for both inputs: preview, preprocessing, classification, verdict, and, for the second input, a heat map. Second, the second decision lies close to the usual 0.5 threshold. A prediction at 56.43 % should be read as "slightly favouring forged" and not as a firm detection, which is why the system displays the confidence and the explanation together instead of a label alone. Third, the Grad-CAM map for that image concentrates on the central body region and largely ignores the ornamented frame. Whether that region contains an alteration cannot be established from the recorded session, because the ground truth of the input is not documented; the image is a screenshot of an artwork, and a "forged" label on it neither confirms nor refutes the detector. A diagnostic of this kind is exactly where a heat map helps: if the highlighted area were a plain background, a reviewer would have good reason to distrust the verdict.

6.1 Design trade-offs

Resolution versus fidelity. Resizing every upload to 224×224 keeps inference fast and matches the backbone, but a high-resolution photograph in which only a few pixels were edited may lose that evidence. **Light denoising.** The small σ preserves residual structure, at the cost of leaving most sensor noise in place. **Compact backbone.** EfficientNet-B0 trades some representational capacity for speed and a small memory footprint, which suits the ordinary laptop hardware listed in Table 1.

VII. PLANNED EVALUATION PROTOCOL

To turn the prototype into a measured system, the following protocol will be applied. It is stated here so that the limits of the current results are explicit and so that later numbers can be compared against a fixed plan.

Data. Authentic and tampered images from the CASIA tampering database [15] will be split into training, validation and test partitions, with the split made at the level of source scenes so that no authentic image and its tampered derivative appear on opposite sides of the split. Pretrained ImageNet weights [16] will be used as the starting point.

Classification metrics. With TP, TN, FP and FN denoting counts for the forged class, together with the area under the ROC curve to remove dependence on τ.

$$\mathrm{Acc} = \frac{TP+TN}{TP+TN+FP+FN}, \qquad F_1 = \frac{2PR}{P+R}, \tag{4}$$

$$P = \frac{TP}{TP+FP}, \qquad R = \frac{TP}{TP+FN}, \tag{5}$$

Robustness. The test set will be re-evaluated after JPEG re-compression at several quality factors, down-scaling, and added Gaussian noise, since these operations are the usual cause of failure of hand-crafted detectors [1, 6].

Ablation. The Gaussian step will be switched on and off to test whether it helps or hurts, and the EfficientNet-B0 backbone will be compared with at least one other backbone.

Explanation quality. For samples with ground-truth masks, the overlap between the thresholded Grad-CAM map M and the true tampered mask T will be measured as IoU = $|M \cap T| / |M \cup T|$.

VIII. APPLICATIONS

The system is intended as an assistive screening tool. In journalism and social-media moderation, it can triage large numbers of images before human fact-checking. In forensic and legal work, a heat map gives an examiner a starting point for closer manual inspection. Banks and identity-verification services can use it as one signal when scanning submitted documents or portraits. Educators and researchers can use the Streamlit/Flask structure as a template for demonstrating forensic models. In every case, the output is advice for a human reviewer and not a verdict.

IX. LIMITATIONS AND FUTURE SCOPE

Limitations. The current evidence is qualitative (Section 6); there are no benchmark figures, and generalisation to manipulations from unseen generators is untested. The confidence value is the network's output probability and is not

guaranteed to be calibrated. Grad-CAM is a coarse, class-discriminative map and does not delineate tampered pixels. The system also decides on a single image and cannot use metadata, provenance or video cues.

Future scope. Planned extensions are (i) running the evaluation protocol of Section 7; (ii) adding generated-image and deepfake data [11] and testing temporal cues for video [12]; (iii) adding a localisation head, in the manner of region-proposal and anomaly-based detectors [5, 10], to produce masks rather than only coarse maps; (iv) calibrating the confidence output; and (v) hardening the deployment with encrypted storage and audit logs for the administrator role.

X. CONCLUSION

This paper presented an AI-based image forgery detection system that combines a compact EfficientNet-B0 classifier, a documented preprocessing chain, Grad-CAM explanations, and a web application with accounts, history and report generation. Recorded sessions demonstrate that the whole pipeline runs from upload to explained verdict, and they also illustrate a limitation that motivates the design: a low-margin decision is only meaningful when accompanied by evidence the user can inspect. The next phase is a quantitative evaluation on public tampering benchmarks under compression and resizing, after which the detector can be judged on measured performance rather than demonstrations.

Acknowledgment

The authors thank Dr. Poornima B, Head of the Department and Program Coordinator, Prof. Puneeth S P, Project Mentor, and the Project Coordinators Dr. Arun Kumar G H and Prof. Santhosh T of the Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, for their guidance during Project Phase 1.

REFERENCES

1. H. Farid, "Image forgery detection," *IEEE Signal Process. Mag.*, vol. 26, no. 2, pp. 16–25, 2009.
2. L. Verdoliva, "Media forensics and DeepFakes: An overview," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 5, pp. 910–932, 2020, doi: 10.1109/JSTSP.2020.3002101.
3. J. Fridrich, D. Soukal, and J. Lukáš, "Detection of copy-move forgery in digital images," in *Proc. Digital Forensic Research Workshop*, Cleveland, OH, USA, Aug. 2003.
4. B. Bayar and M. C. Stamm, "A deep learning approach to universal image manipulation detection using a new convolutional layer," in *Proc. 4th ACM Workshop Inf. Hiding Multimedia Security (IH&MMSec)*, 2016, doi: 10.1145/2909827.2930786.
5. P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Learning rich features for image manipulation detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 1053–1061.
6. M. Zanardelli, F. Guerrini, R. Leonardi, and N. Adami, "Image forgery detection: A survey of recent deep-learning approaches," *Multimedia Tools Appl.*, vol. 82, pp. 17521–17566, 2023, doi: 10.1007/s11042-022-13797-w.
7. R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "DeepFakes and beyond: A survey of face manipulation and fake detection," *Inf. Fusion*, vol. 64, pp. 131–148, 2020, arXiv:2001.00179.
8. D. Cozzolino, G. Poggi, and L. Verdoliva, "Splicebuster: A new blind image splicing detector," in *Proc. IEEE Int. Workshop Inf. Forensics Security (WIFS)*, 2015, pp. 1–6, doi: 10.1109/WIFS.2015.7368565.
9. B. Bayar and M. C. Stamm, "Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2691–2706, Nov. 2018, doi: 10.1109/TIFS.2018.2825953.
10. Y. Wu, W. AbdAlmageed, and P. Natarajan, "ManTra-Net: Manipulation tracing network for detection and localization of image forgeries with anomalous features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 9543–9552.
11. A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 1–11, doi: 10.1109/ICCV.2019.00009.
12. R. Caldelli, L. Galteri, I. Amerini, and A. Del Bimbo, "Optical flow based CNN for detection of unlearnt deepfake manipulations," *Pattern Recognit. Lett.*, vol. 146, pp. 31–37, 2021.
13. M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, ser. PMLR, vol. 97, 2019, pp. 6105–6114.
14. R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, 2020, doi: 10.1007/s11263-019-01228-7.
15. J. Dong, W. Wang, and T. Tan, "CASIA image tampering detection evaluation database," in *Proc. IEEE China Summit Int. Conf. Signal Inf. Process. (ChinaSIP)*, Beijing, China, 2013, pp. 422–426, doi: 10.1109/ChinaSIP.2013.6625374.
16. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.
17. M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implementation (OSDI)*, 2016, pp. 265–283.