IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026

DOI: 10.17148/IJARCCE.2026.1510156

AI-BASED SKIN LESION CLASSIFICATION AND REPORTING USING DEEP LEARNING

SharanaBasavaRaddi B[1], Nirmitha P[2], Sushmitha H[3], Chinmai B[4],Thara R[5]

Assistant Professor, Department of Computer Science and Engineering, S J M Institute of Technology, Chitradurga[1]

B.E Student, Department of Computer Science and Engineering, S J M Institute of Technology, Chitradurga[2,3,4,5]

Abstract: Skin conditions can have similar visual characteristics, making preliminary assessment from images challenging. This paper presents a web-based Skin Disease Detector that combines a trained TensorFlow/Keras image-classification model with a Flask application. A user can create an account, upload a skin image, and receive three ranked class predictions with confidence scores. The system resizes images to 300 × 300 pixels, applies EfficientNet-compatible preprocessing, and passes the processed image to the loaded model. Results are stored in an SQLite database and can be revisited through prediction history. The application also provides disease information and downloadable PDF reports. The work demonstrates an end-to-end software implementation for educational and preliminary screening support. The supplied project files do not include a documented test dataset or evaluation metrics; therefore, no claim of diagnostic accuracy or clinical validation is made. The tool is not a substitute for examination by a qualified healthcare professional.

Index Terms: deep learning, EfficientNet, image classification, skin lesions, web application

I. INTRODUCTION

Skin lesion assessment often depends on visual examination and, where indicated, further clinical investigation. Image-classification methods can help organize visual information, but their output must be interpreted carefully because image quality, lesion appearance, and differences between patient populations can affect predictions. Convolutional neural networks are widely used for image recognition, while EfficientNet introduces a balanced approach to scaling network depth, width, and input resolution [1]. Research datasets such as HAM10000 have also supported the study of automated skin-lesion classification, while illustrating the importance of documenting image sources and diagnostic labels [2].

This project implements a browser-based application that connects image upload, model inference, user-specific history, disease information, and PDF report generation. Its contribution is primarily system integration: it places a trained model inside a usable web workflow instead of presenting classification as an isolated script. The project overview identifies seven output classes, including melanoma, melanocytic nevi, basal cell carcinoma, benign keratosis-like lesions, actinic keratoses, dermatofibroma, and vascular lesions. The present manuscript describes the software design and current capabilities, not a clinical performance study.

II. SYSTEM DESIGN AND METHODOLOGY

The application is implemented in Python using Flask for request handling and page rendering, Flask-Login for session management, Flask-SQLAlchemy for database access, TensorFlow/Keras for model inference, and Pillow/NumPy for image preparation. SQLite stores account and prediction records. The interface is organized around registration, login, a dashboard, image prediction, disease details, prediction history, analysis summaries, and report download.

When a user submits an image, the server checks the file extension against the permitted PNG, JPG, JPEG, and GIF formats, creates a sanitized filename, and saves the upload. The image is loaded and resized to 300 × 300 pixels, converted to an array, expanded to a batch, and passed through the preprocessing function associated with EfficientNet. The loaded model produces class scores; the application sorts these scores and displays the three highest-ranked labels with their corresponding values. These values are model confidence scores and should not be interpreted as calibrated clinical probabilities.

III. SYSTEM DESIGN AND METHODOLOGY (CONTINUED)

After inference, the application stores the leading predicted class, its confidence value, the full top-three output, the image filename, the user identifier, and a timestamp in the prediction-history table. A user can later view earlier results, open the associated disease-information page, or request a PDF report. The database models separate user records from prediction records, and each prediction is linked to the account that submitted it. Uploaded-image access is routed through an authenticated endpoint with a filename ownership check.

The disease-information module supplies short descriptions and additional details for supported labels. The report generator uses ReportLab to create a downloadable document containing the predicted condition, report information, and related disease content. The analysis page summarizes the user's saved predictions by class and calculates mean displayed

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 10, October 2026
DOI: 10.17148/IJARCCE.2026.1510156

confidence for each class. These are application-level summaries; they do not measure model accuracy because no ground-truth labels are compared with predictions in the supplied implementation.

IV. IMPLEMENTATION AND OBSERVATIONS

The submitted source code supports an end-to-end workflow from account creation to image-based inference and retrieval of past results. The prediction function explicitly selects the three largest model outputs, while the history feature makes previous analyses available to the same authenticated user. PDF export provides a portable summary that may be useful for discussion with a clinician. The use of common Python libraries also makes the application relatively straightforward to run in a local development environment.

The current evidence is implementation-based. The supplied files do not provide the model weights, class-label file, training configuration, dataset provenance, independent test split, confusion matrix, sensitivity, specificity, or other measured evaluation results. Consequently, it would be misleading to report an accuracy percentage, compare the model with dermatologists, or claim that the system improves clinical outcomes. A proper evaluation should use a documented, appropriately licensed dataset, separate training and test data at the patient level where possible, and report per-class precision, recall, F1-score, and a confusion matrix. Performance should also be examined across image sources and skin tones before any clinical use.

Several safeguards are important before deployment. The application's development configuration contains a placeholder secret key and should be changed to an environment-managed secret. Production deployment should use HTTPS, stronger upload validation, size and content checks, controlled retention of sensitive images, access logging, and a production WSGI server. Medical images and predictions should be handled as sensitive information. The interface should clearly explain uncertainty and encourage professional assessment, particularly for a changing, bleeding, painful, or otherwise concerning lesion.

V. LIMITATIONS

The model is loaded from an external project path, and the supplied materials do not establish its training history or generalization performance. The displayed class scores may vary with image quality and may not reflect real-world disease likelihood. The current repository also does not include dedicated automated tests or a documented clinical evaluation. The system should therefore be treated as a prototype for educational and research use rather than a diagnostic device.

VI. CONCLUSION AND FUTURE WORK

This paper presented a web-based skin-lesion classification application that combines a TensorFlow/Keras model with account management, image uploads, ranked predictions, disease descriptions, prediction history, summary analysis, and PDF reporting. The implementation provides a practical structure for connecting a deep-learning component to a user-facing workflow. However, software functionality alone does not establish diagnostic reliability. Future work should document the model architecture and training data, conduct independent and clinically meaningful evaluation, test robustness across image conditions and skin tones, improve explainability, and strengthen privacy and deployment security. Clinical decisions must remain with qualified healthcare professionals; the application should be used only as an aid for preliminary information.

REFERENCES

[1] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” in Proceedings of the 36th International Conference on Machine Learning, vol. 97, pp. 6105–6114, 2019. Available: https://proceedings.mlr.press/v97/tan19a.html

[2] P. Tschandl, C. Rosendahl, and H. Kittler, “The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions,” Scientific Data, vol. 5, article 180161, 2018. doi: 10.1038/sdata.2018.161.

[3] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1251–1258, 2017.

 This work is licensed under a Creative Commons Attribution 4.0 International License