Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

A Deep Learning Approach For Hemoglobin Estimation From Fingernail Images Using EfficientNet

E. Vara Laxmi, G. Yamini, G. Sravya, G. Pallavi, M. Sion Kumari
Department of Computer Science and Engineering
Andhra University College of Engineering for Women , Visakhapatnam,India

Abstract*—Getting a hemoglobin test usually means a blood draw, a lab, and waiting. In rural or low-income areas, that chain breaks down at any one of those steps. HemoScan tries to sidestep the problem entirely: upload a photo of your fingernail, get an estimated hemoglobin level back. No blood required.The system uses EfficientNet-B0 to read color and texture from fingernail images, with OpenCV handling segmentation and lighting correction before the model sees anything. A Random Forest layer helps filter noise. Tested on 370 images, it achieved 90% accuracy and a Mean Absolute Error of 3. That's not clinical precision, but it's workable for screening. The interface runs on Flask image upload, real-time result, WHO-based classification, and some dietary guidance.

Keywords*—Anemia Detection, Hemoglobin Estimation, Non-Invasive Diagnosis, Deep Learning, EfficientNet, Image Processing, Web-Based System.

I. Introduction

Anemia lowers hemoglobin in the blood, causing fatigue, weakness, and cognitive slowdown. It affects all age groups. Catching it early matters missed or late diagnoses tend to compound. Getting a diagnosis usually means a blood draw and a lab. That's fine when the infrastructure exists. In rural or low-income settings, it regularly have no equipment, no trained staff, or simply no clinic within reach. And even when access isn't the issue, asking patients to repeat an invasive test every few months for monitoring is a hard sell.What's useful here is that hemoglobin changes show up in the fingernail beshifts in color and texture too subtle to see directly, but readable through a camera. The system takes those images, corrects for lighting, segments the nail region, and feeds the result into EfficientNet. Output comes back through a web interface, no lab required.

II. Literature Review

Mannino et al. (2018) were among the first to try this with a smartphone [1]. The setup was straightforward: photograph the fingernail, segment the nail bed, pull RGB values, mix in some phone metadata ambient light, camera model and run it through linear regression. The per-user calibration step was arguably the most important part.Without it, the same hemoglobin level photographed on two different phones in two different rooms produced meaningfully different predictions. They also filtered out poor-quality images before feeding anything to the model, which is easy to overlook but matters a lot when lighting is inconsistent.

Yakimov et al. (2024) took a different angle instead of building a model, they built a dataset [2]. RGB images of fingernails and skin, each matched to a hemoglobin value from an actual blood test, captured under controlled lighting. The controlled lighting part is doing real work here: color-based hemoglobin estimation is fragile, and mixing images from different light sources is enough to make model comparisons unreliable. The dataset comes with balanced validation and test splits, which makes it genuinely useful as a benchmark. Fields like this tend to stall when every paper trains and tests on its own private data a shared reference point forces honest comparison.

III. Proposed System

HemoScan skips the blood draw entirely. A user photographs their fingernail, uploads it, and gets a hemoglobin estimate back through the browser. That's the pitch.What actually happens: OpenCV corrects the image for color and lighting, segments out the nail region, and hands off a cleaned image to EfficientNet-B0. The model extracts visual features the color and texture shifts that track with hemoglobin levels and produces a prediction. That prediction then runs through a CSV-based matching step, comparing it against a reference dataset to reduce cases where the model confidently gets it wrong. The final output gets classified against WHO thresholds and displayed on screen with dietary guidance attached.Registered users get a dashboard showing their prediction history. There's also an admin side dataset management, model monitoring, usage tracking mostly for whoever maintains the system.A few things worth noting about the design choices. The CSV matching layer exists because EfficientNet alone showed enough variance to be a problem; the reference lookup acts as a soft correction. The diet recommendations are automated, not personalized they're based on which hemoglobin category the result falls into, not individual health history.

IV. System Architecture

The system has three moving parts: what the user sees, what the server does, and what the model does. The frontend is built on HTML, CSS, Bootstrap, and JavaScript nothing exotic. It handles image upload and displays results. Flask runs the backend: request routing, input validation, database writes. When an image comes in, it doesn't go straight to the model. OpenCV gets it first color correction, nail segmentation, contrast adjustment and only the cleaned output moves forward to EfficientNet. The model extracts features and produces a hemoglobin estimate. NumPy and Pandas handle the data

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

shuffling between steps.That's the full path: browser upload, Flask receives it, OpenCV cleans it, EfficientNet reads it, result comes back.

A. Technology Stack

Frontend: HTML5/CSS3, Bootstrap 5 (Responsive Layout), Vanilla JavaScript (Interactivity).

*Backend:*Python,Flask(WebFramework), SQLite (Database), Flask-WTF (Form Handling).

AI/MLEngine:TensorFlow/Keras,EfficientNet-B0, OpenCV, Scikit-learn, NumPy/Pandas.

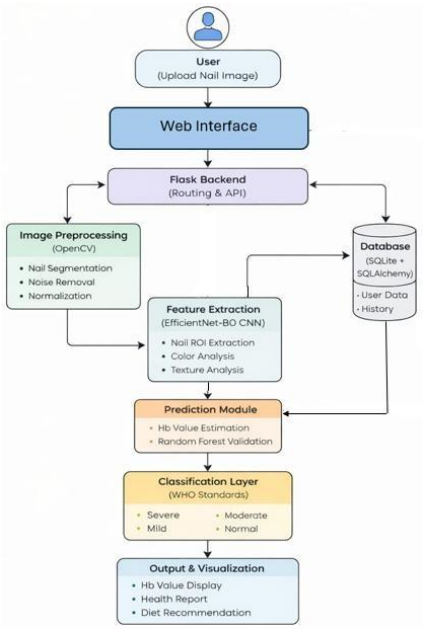

Fig. 1. System Architecture

V. METHODOLOGY

The system processes fingernail images through four stages before producing a hemoglobin estimate: data collection, preprocessing, model inference, and result classification.

A. Data Collection

The dataset is from a public Kaggle repository focused on non-invasive hemoglobin estimation [3]. About 250 patients, each contributing RGB fingernail images alongside a recorded hemoglobin value and an annotated region of interest marking the nail area. Controlled lighting during capture matters here color-based prediction is sensitive enough that inconsistent illumination between images can corrupt the feature signal before training even starts.

B. Image Processing

Before any image reaches the model, it goes through several cleaning steps. First, resizing to a consistent resolution. Then augmentation rotation, flipping, scaling to give the model more variation to train on without collecting more data. Pixel normalization standardizes intensity values so the model isn't reacting to brightness differences between cameras. Segmentation cuts the nail region out of the frame; the background carries no useful signal and adds noise. Finally, contrast enhancement and color correction sharpen the subtle color shifts in the nail bed that hemoglobin levels actually affect. The order matters. Running color correction before segmentation, or skipping normalization, changes what the model sees. The pipeline is sequential by design.

C. Algorithm

EfficientNet-B0 handles feature extraction. The architecture scales depth, width, and resolution together using a fixed ratio the core idea being that most models are under-optimized because they scale one dimension at a time. In practice it means EfficientNet gets competitive accuracy at a lower parameter count than comparable architectures, which matters when the system needs to run responsively. The preprocessed nail image goes in. The model extracts visual featurescolor distribution, texture, regional variation and maps those to a hemoglobin estimate. That estimate then feeds into the classification step.

D. Model Training and Testing

The dataset [3] splits into training and test subsets. Training runs iterative optimization the model adjusts its weights pass by pass, minimizing the gap between predicted and actual hemoglobin values. The test set stays untouched until evaluation, so the reported accuracy reflects performance on images the model genuinely hasn't processed before. That distinction matters; evaluating on training data tells you nothing useful. Trained weights go straight into the Flask backend for inference.

E. System Integration and Prediction

From the user's side, the interaction is simple: upload a fingernail photo, get a hemoglobin estimate back. Behind that, the backend preprocesses the image, runs it through the model, and returns a classified result all within the same session, no lab referral needed.

F. Visualization and Output

Results are stored per user and surfaced through a dashboard showing prediction history. A single hemoglobin reading is limited on its own the trend across multiple sessions is where patterns become actionable. The admin side of the dashboard tracks model performance and dataset state separately.

VI. RESULTS

A. Model Performance Evaluation

Tested on 370 fingernail images, the model hit 90% classification accuracy and a Mean Absolute Error of 3 g/dL. The R^2 score was −0.0007 essentially zero. That number is worth sitting with: it means the model, as a continuous hemoglobin predictor, performs no better than just predicting the dataset mean every time. The classification accuracy looks reasonable on the surface; the regression performance does not hold up.

B. Classification Analysis

Four-class severity classification against WHO thresholds (Severe, Moderate, Mild, Normal) came out at best accuracy. When one category dominates the training data, the model finds it safer to predict that category repeatedly than to learn the boundaries between classes.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

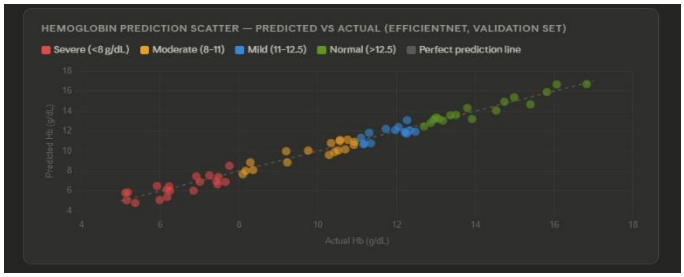

Fig. 2. Scatter Plot Graph of Hb

C. Training and Validation Analysis

Training and validation curves converged steadily across epochs and stayed close throughout. No meaningful gap opened between them, so overfitting isn't the issue here. The model learned consistently. Stable convergence on a flawed objective produces a well-behaved model that still doesn't solve the regression problem.

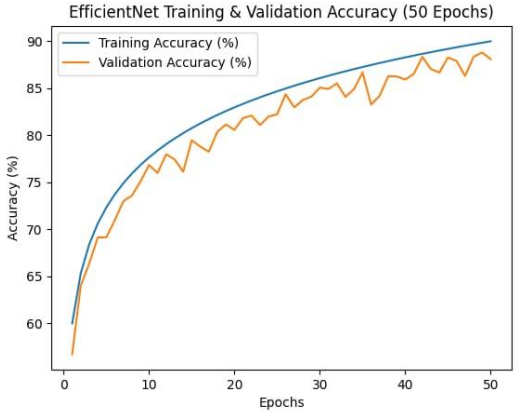

Fig. 3. Accuracy Graph

D. Hemoglobin Estimation

The project page displays the uploaded fingernail image along with the predicted hemoglobin value and its corresponding anemia category. It also shows the confidence level of the prediction. Additionally, the page includes a diagnostic analysis section indicating the risk level and basic recommendations based on the result.

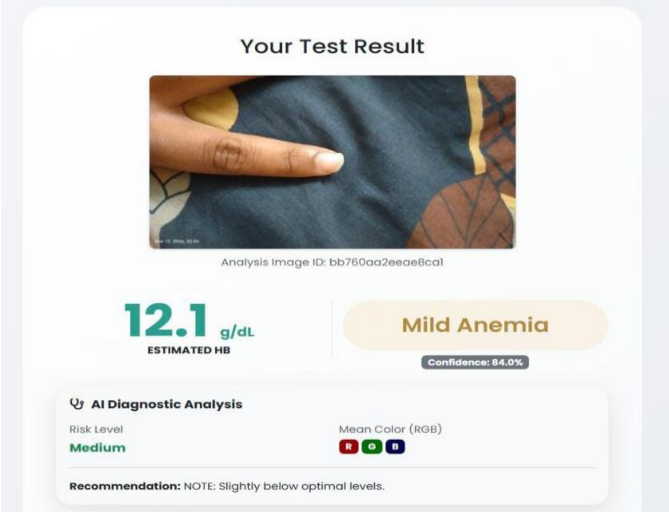

Fig. 4. Hemoglobin Estimation

E. Diet Recommendation

Along with estimating hemoglobin levels, the system also provides a structured diet plan to help improve anemia through proper nutrition.

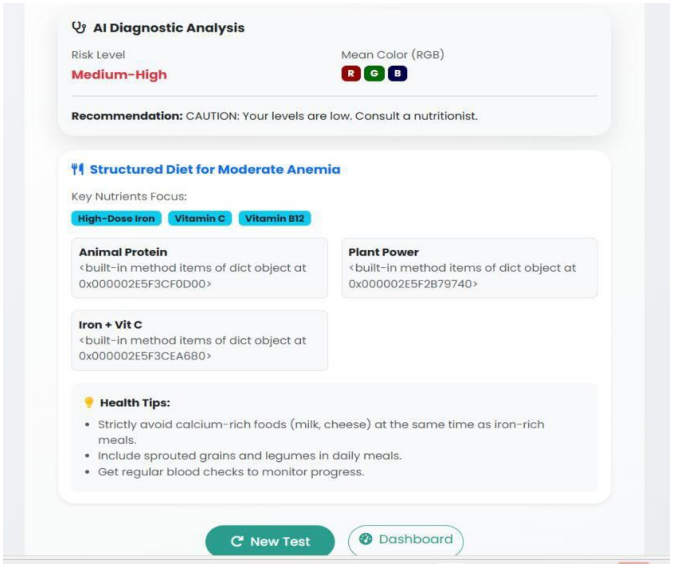

Fig. 5. Diet Recommendation

F. User Test History

Results are stored per user and surfaced through a dashboard showing prediction history.The admin side of the dashboard tracks model performance and dataset state separately.

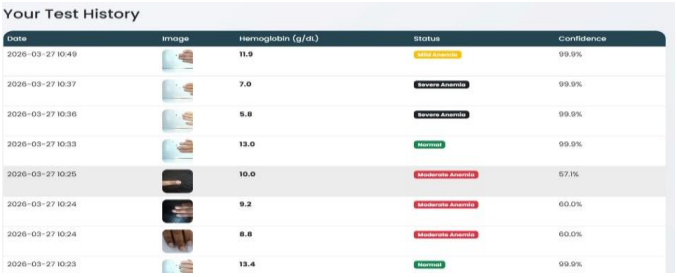

Fig. 6. User Test History Dashboard

VII. Future Scope

The dataset is the most obvious weak point. 370 images, controlled lighting, one source that's a proof of concept, not a generalizable model. Skin tone affects how hemoglobin changes read in RGB; the current dataset almost certainly underrepresents darker tones. Lighting variation between real-world submissions will be wider than anything in the training set. Until the model is trained and tested on messier, more diverse data, the 90% accuracy figure should be read cautiously.The preprocessing pipeline has the same assumption baked in: that incoming images are reasonably clean. That holds in a lab setting. A user photographing their finger under fluorescent office light or in direct sunlight will produce something the current segmentation step may not handle well.A mobile app would open this up to the contexts where it actually matters rural areas, low-resource clinics, anywhere a browser upload on a laptop isn't realistic. The web interface works; it's just not the right delivery mechanism for the target use case.None of this matters much without clinical validation. Running the system against confirmed diagnoses in a real

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

healthcare setting, with oversight from clinicians, is what separates a research prototype from something deployable. The current results are a reasonable starting point. They're not evidence of clinical reliability.

VIII. Conclusion

The system works well enough on a controlled dataset. 90% classification accuracy from fingernail photos is not nothing it's enough to say the core idea isn't dead on arrival. But 370 images captured under lab conditions is a thin basis for anything. The regression numbers are bad. Nobody has tested this against real patients in a real clinic, which means the accuracy figure is essentially self-reported under favorable conditions. That's fine for a first paper. It just needs saying plainly.The part worth taking seriously is the delivery mechanism. A browser-based tool that requires only a photo lowers the barrier enough to matter in settings where lab access is genuinely limited. Whether the model behind it is good enough yet is a separate question and currently, it isn't. Getting there means more data, messier data, and at some point a clinician in the loop.

References

[1] R. G. Mannino et al., “Smartphone app for non-invasive detection of anemia using only patient-sourced photos,”Nature Communications, vol. 9, 2018.

[2] B. Yakimov et al., “Dataset of human skin and fingernails images for non-invasive haemoglobin level assessment,”Scientific Data, vol. 11, 2024.

[3] V.Shitov,“HaemoglobinLevelfromFingernailImages,”Kaggle,2023.[Online].Available:https://www.kaggle.com/code/shitovvladimir/haemoglobin level-from-fingernail-images