International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 06 | Jun 2026 www.irjet.net p-ISSN: 2395-0072

Intelligent Dermatology System for Nail Disease Detection Using CNN

V. Raveendra[1], K. Ramya Keerthi[2]

[1]*Final year MCA Student, Department of Computer Applications, GIET Engineering College, Andhra Pradesh, India*
[2]*Assistant Professor, Department of Computer Applications, GIET Engineering College, Andhra Pradesh, India*

Abstract - *Nail health is often overlooked, even though visible changes in nails can indicate infections, nutritional issues, or other medical conditions. This project, "Nail Disease Classification and Prediction System", is designed to support early identification of common nail diseases using image-based deep learning. The system allows a user to upload a nail image and receive a predicted disease category with a confidence score. The application is developed as a Django-based web platform with separate user and admin modules. Users can register, log in after admin approval, train the model, and perform nail disease prediction. The admin module manages registered users and activates valid accounts. The machine learning component uses an EfficientNet-B0 based classification model with image preprocessing, data augmentation, class balancing, and fine-tuning to improve prediction performance. The trained model is evaluated using validation accuracy, loss, F1-score, and confusion matrix visualization. The dataset is organized into training and validation folders containing different nail disease classes such as healthy nails, clubbing, pitting, blue finger, onychogryphosis, and acral lentiginous melanoma. After training, the system stores the model, class labels, metrics, and visual results for future prediction. This project is mainly intended as an educational and research-based tool that demonstrates how artificial intelligence can assist in healthcare-related image classification. It is not a replacement for professional medical diagnosis, but it can provide a useful first-level screening approach and encourage timely medical consultation.*

Key Words: Nail Disease Classification, Nail Disease Prediction, Deep Learning, Image Classification, Medical Image Analysis, Django Web Application, PyTorch, EfficientNet-B0, Transfer Learning, Convolutional Neural Network, Image Preprocessing, Data Augmentation, GPU Training, Disease Detection, Healthcare AI, Nail Image Dataset, Confusion Matrix, F1-Score, Validation Accuracy, Web-Based Diagnosis Support.

1. INTRODUCTION

Nail problems are usually treated as small health issues, but in some cases they can give early signs of a disease or infection. A change in nail colour, shape, thickness, or texture may look simple from outside, but it can still indicate an underlying condition. Many people do not visit a doctor for nail-related problems at the early stage because they may not know whether the condition is serious or not. This project is developed with the idea of giving users a quick and basic prediction about nail diseases by using an image of the nail.

The "Nail Disease Classification and Prediction System" is a web-based application that combines deep learning with a simple user interface. The user can upload a nail image, and the system predicts the possible disease category based on the trained model. The application is not designed to give final medical advice, but it can help users understand the condition and decide whether they should consult a medical professional.

This project is implemented using Django for the web application and a deep learning model for image classification. The system has user and admin modules. A new user can register, but the account becomes active only after admin approval. After login, the user can access the dashboard, train the model, and use the prediction feature. The admin can view registered users and activate their accounts.

For classification, the system uses an EfficientNet-B0 based model. The dataset contains nail images arranged into different categories such as healthy nail, pitting, clubbing, blue finger, onychogryphosis, and acral lentiginous melanoma. Before training, images are resized and processed so that the model can learn useful visual patterns. Techniques such as augmentation, class balancing, and fine-tuning are used to improve the training process.

After training, the model is saved and used for future predictions. The system also generates useful evaluation results such as validation accuracy, loss, F1-score, training graph, and confusion matrix. These outputs help to understand the performance of the model.

Overall, this project shows how artificial intelligence can be used in a practical healthcare-related application. It provides a simple way to identify possible nail disease categories from images and creates awareness about nail health. Although it cannot replace a doctor, it can act as a first-level support system for basic screening and learning purposes.

2. METHODOLOGY

The methodology of this project follows a step-by-step process, starting from dataset preparation and ending

with disease prediction through the web application. The system is designed in such a way that both the web part and the machine learning part work together to provide a smooth prediction process for the user.

First, the nail disease dataset is collected and arranged into separate folders based on disease categories. The dataset is divided into training and validation folders. Each folder contains class-wise image folders such as healthy nail, pitting, clubbing, blue finger, onychogryphosis, and acral lentiginous melanoma. This folder structure helps the model automatically identify class labels during training.

After dataset preparation, image pre-processing is performed. Since images may have different sizes and qualities, all images are resized into a fixed input size of `224 x 224`. The images are then converted into tensor format and normalized. During training, augmentation techniques such as horizontal flipping, rotation, color adjustment, and affine transformation are applied. These techniques help the model learn better from variations in nail images and reduce over fitting.

The classification model is built using an EfficientNet-B0 based deep learning architecture. EfficientNet-B0 is selected because it gives good performance for image classification while being lightweight compared to many large deep learning models. In this project, transfer learning is used. The model first uses pretrained image features and then learns nail-disease-specific patterns from the project dataset.

Training is carried out in two stages. In the first stage, the main feature extraction layers are frozen, and only the classifier part is trained. This helps the model adjust to the nail disease classes without disturbing the pretrained features. In the second stage, selected upper layers of the model are unfrozen and fine-tuned with a smaller learning rate. This improves the model's ability to recognize disease-specific details.

To handle imbalance in the dataset, class weights are calculated and applied during training. This ensures that classes with fewer images are also given importance. The model is trained using cross-entropy loss with label smoothing, which helps avoid overconfidence in predictions. If a compatible NVIDIA GPU is available, the training process runs on GPU using CUDA, which reduces training time.

After training, the best-performing model is saved and the system also saves class labels and training results. Evaluation is done using validation accuracy, validation loss, macro F1-score, weighted F1-score, and confusion matrix. Training graphs and confusion matrix images are generated and displayed on the training page.

The web application is developed using Django. It contains user and admin modules. A user can register, but the account is activated only after admin approval. Once activated, the user can log in, train the model, and upload nail images for prediction. The admin can log in, view registered users, and activate valid accounts.

For prediction, the uploaded image is first checked for valid file type. Then it is resized, normalized, and passed to the trained model. The model returns probability values for each disease class. The class with the highest probability is selected as the final prediction, and the confidence score is shown to the user. The system also displays top prediction results to give better understanding of the output.

Thus, the complete methodology includes dataset organization, pre-processing, model training, evaluation, model saving, web integration, and prediction. This approach makes the project useful as a basic AI-based nail disease classification system for educational and screening purposes.

2.1 Data Collection and Pre-processing

For this project, nail images are used as the main input data. The dataset contains images of different nail conditions and is arranged class-wise so that the model can learn each disease category separately. The images are stored inside the `media/data` folder of the project. The dataset is divided into two main parts: training data and validation data.

The training images are stored in `media/data/train`, and the validation images are stored in `media/data/validation`. Inside these folders, images are grouped into separate class folders. The classes used in this project include healthy nail, pitting, clubbing, blue finger, onychogryphosis, and acral lentiginous melanoma. This folder-based arrangement makes it easier for the deep learning model to automatically read the images along with their correct labels.

Before giving the images to the model, preprocessing is carried out. Since the collected images may not have the same size, every image is resized to `224 x 224` pixels. This fixed size is required because the EfficientNet-B0 model expects images in a consistent input format. After resizing, the images are converted into tensor format so that they can be processed by the deep learning model.

Normalization is also applied to the images. In this step, pixel values are adjusted using standard mean and standard deviation values. This helps the model learn more smoothly and reduces the effect of brightness or color differences between images. Normalization also

makes the input format suitable for the pretrained EfficientNet-B0 model used in the project.

During training, data augmentation is applied to increase variation in the dataset. The project uses transformations such as horizontal flipping, small rotation, color adjustment, and affine transformation. These changes create slightly different versions of the same image during training. This helps the model understand nail patterns better and reduces the chance of memorizing only the training images.

The dataset may not contain the same number of images for every class. Because of this, class balancing is used during training. Class weights are calculated based on the number of images in each category. Classes with fewer images are given more importance so that the model does not become biased toward classes with more images.

For validation, the images are only resized, converted to tensor format, and normalized. Augmentation is not applied to validation images because validation data should represent real testing conditions. This helps in checking how well the trained model performs on images it has not learned from directly.

In short, the data collection and pre-processing stage includes collecting nail images, arranging them into class-wise folders, splitting them into training and validation sets, resizing, normalization, augmentation, and class balancing. These steps prepare the dataset properly before training and help the model produce more reliable predictions.

2.2 The Core Classification Model (EfficientNet-B0)

The core classification model used in this project is "EfficientNet-B0", implemented with "PyTorch and torchvision". EfficientNet-B0 is a convolutional neural network model mainly used for image classification tasks. It is selected for this project because it gives a good balance between accuracy and model size, which makes it suitable for a web-based nail disease prediction system.

In this project, EfficientNet-B0 works as the main feature extractor. The model takes a nail image as input and learns visual patterns such as nail color, surface texture, shape changes, thickness, and abnormal markings. These features are then used to classify the image into one of the nail disease categories.

The project uses "transfer learning", where the EfficientNet-B0 model is loaded with pretrained ImageNet weights. This means the model already has general image understanding before it is trained on the nail disease dataset. Instead of learning everything from the beginning, it adapts its existing knowledge to identify nail disease patterns.

The original classifier layer of EfficientNet-B0 is replaced with a custom classifier suitable for this project's classes. The custom classifier contains dropout layers, a fully connected layer, SiLU activation, and a final output layer. The final layer produces probability scores for each nail disease class.

Training is done in two stages. In the first stage, the backbone layers are frozen and only the classifier part is trained. In the second stage, the top layers of the EfficientNet model are unfrozen and fine-tuned with a smaller learning rate. This helps the model improve its understanding of nail-specific features without losing the useful pretrained features.

The model uses cross-entropy loss with label smoothing and class weights. Class weights help when some nail disease classes have fewer images than others. Label smoothing helps the model avoid becoming too confident in one prediction. If an NVIDIA GPU is available, the model training runs using CUDA, which makes the training process faster.

Overall, EfficientNet-B0 is the main deep learning model in this project. It classifies uploaded nail images into disease categories and returns the predicted class along with a confidence score.

2.3 The Styling Recommendation Engine

The main intelligent part of this project is the Nail Disease Prediction Engine. It is responsible for analyzing the uploaded nail image and predicting the possible disease category. When a user uploads an image, the system first pre-processes it by resizing and normalizing it. After that, the processed image is passed into the trained EfficientNet-B0 model.

The model studies the visual features of the nail, such as color changes, nail shape, texture, thickness, and visible abnormal patterns. Based on these features, it gives probability scores for each disease class. The class with the highest score is selected as the final prediction, and the confidence value is shown to the user.

This engine works as the connection between the web application and the deep learning model. The Django application handles the user interface and image upload, while the prediction engine handles image processing, model loading, classification, and result generation. In this way, the system provides a simple and practical method for identifying possible nail disease categories from images.

2.4 System Integration and Deployment

System integration in this project means connecting the web application, database, dataset, and machine learning model into one working system. The project is developed using Django, so the frontend pages, backend logic, user management, admin control, model training, and prediction process are all handled inside the same web application.

The frontend part of the system is created using HTML, CSS, Bootstrap, and Django templates. These pages allow users to register, log in, train the model, and upload nail images for prediction. The backend part is handled through Django views. When a user performs an action on the website, the request is sent to the correct Django view, and that view decides what operation should be performed.

The user module is connected with the SQLite database. When a new user registers, the details are validated through the registration form and saved in the database. The password is stored in hashed format for better security. The admin module is also connected with the same database so that the admin can view registered users and activate valid accounts. Only activated users are allowed to log in and use the main features of the system.

The machine learning module is integrated with the Django user module. When the user clicks the training option, the Django view calls the training function from the machine learning pipeline. The system loads images from the training and validation folders, trains the EfficientNet-B0 model, evaluates the result, and stores the trained model as `best_model.pth`. It also saves class labels, metrics, training graph, and confusion matrix. These outputs are later shown on the training page.

For prediction, the uploaded nail image is passed from the Django view to the prediction function. The image is preprocessed and given to the trained model. The model returns the predicted class and confidence score. The result is then sent back to the webpage and displayed to the user in a simple format.

Deployment in this project is done locally using Django's development server. After installing all required packages, migrations are applied to prepare the database. Then the project is started using the `python manage.py runserver` command. Once the server starts, the system can be accessed in a browser using `http://127.0.0.1:8000/`.

The project also supports GPU-based training when a compatible NVIDIA GPU and CUDA-supported PyTorch installation are available. During training, the system automatically checks whether GPU is available. If GPU is found, the model is trained using CUDA; otherwise, it continues on CPU. This makes the system flexible for both normal computers and GPU-supported machines.

Overall, the integration combines Django, SQLite, PyTorch, dataset folders, trained model files, and frontend templates into a single working application. This allows the user to move from registration to prediction through one complete system.

3. DISCUSSION

The Nail Disease Classification and Prediction System shows how image-based deep learning can be used in a simple healthcare-related application. The project combines a Django web application with a deep learning model, so the user does not need to interact with code directly. From the user's side, the process is simple: register, log in after approval, upload a nail image, and view the predicted result. Behind this simple flow, the system performs image preprocessing, model loading, classification, and confidence calculation.

One important part of this project is the use of EfficientNet-B0 as the classification model. Since nail disease images contain small visual differences, the model must learn details such as color variation, nail surface texture, shape, and abnormal patterns. EfficientNet-B0 is useful here because it is lightweight and still performs well for image classification. The use of transfer learning also helps because the model does not need to learn image features completely from the beginning.

The training process includes augmentation, class balancing, and fine-tuning. These steps are useful because real-world images are not always captured in the same lighting, angle, or background. Augmentation helps the model become more flexible by showing it different versions of the training images. Class balancing is also important because some disease categories may have fewer images than others. Without balancing, the model may give more importance to classes with more samples.

The system also generates validation accuracy, loss, F1-score, training graph, and confusion matrix. These outputs are useful because accuracy alone does not always explain the real performance of a model. For example, if one class has many more images than another class, the model may still show good accuracy while performing poorly on smaller classes. The confusion matrix helps identify where the model is making mistakes between similar-looking classes.

The web application part makes the project more practical. Instead of keeping the model only as a script or notebook, it is connected with a user interface. The admin activation process adds a basic level of control over who can access the system. The prediction page allows users to

upload images and receive results in a readable format. This makes the project suitable for demonstration and educational use.

At the same time, the system has some limitations. The prediction result depends heavily on the quality and size of the dataset.

Another point is that nail diseases can sometimes look similar to each other. A model may confuse classes when the visual difference is very small. In real medical practice, doctors may consider additional information such as symptoms, duration, patient history, and physical examination. This project uses only image input, so its prediction is limited to visual patterns.

Overall, the project successfully demonstrates the integration of deep learning with a web-based platform for nail disease classification. It shows how artificial intelligence can assist in early awareness and screening. With a larger and more diverse dataset, improved validation, and medical expert verification, the system can be further improved in the future.

4. CONCLUSIONS

The Nail Disease Classification and Prediction System was developed to show how deep learning can be used for identifying nail disease categories from images. The project combines a Django web application with an EfficientNet-B0 based classification model, making the system easy to access through a browser instead of running the model manually through code.

Through this system, users can register, log in after admin approval, train the model, and upload nail images for prediction. The admin module provides basic control by allowing the admin to view and activate registered users. The machine learning module handles image preprocessing, model training, evaluation, and prediction. After training, the system saves the trained model, class labels, metrics, and visual outputs such as training graphs and confusion matrix.

The project demonstrates that image classification techniques can be applied to healthcare-related problems in a practical way. By using transfer learning, augmentation, class balancing, and fine-tuning, the model is able to learn useful patterns from nail images. The prediction output gives the disease class along with a confidence score, which helps users get a basic idea about the uploaded nail image.

However, the system should not be considered a replacement for medical diagnosis. Nail diseases can sometimes look similar, and the final decision should always be made by a qualified medical professional. The accuracy of the system also depends on the quality, size, and variety of the dataset used for training.

Overall, this project provides a useful demonstration of combining artificial intelligence and web technology for nail disease classification. It can be improved further by using a larger dataset, adding more disease categories, improving model accuracy, and including expert medical validation.

REFERENCES

[1] A. J. Wulkan and A. Tosti, “Pediatric nail conditions,” Clinics in dermatology, vol. 31, no. 5, pp. 564–572, 2013.

[2] J. Velasco, C. Pascion, J. W. Alberio, J. Apuang, J. S. Cruz, M. A.Gomez, B. Molina Jr, L. Tuala, A. Thio-ac, and R. Jorda Jr, “A smartphone-based skin disease classification using mobilenet cnn,” arXiv preprint arXiv:1911.07929, 2019.

[3] Z. Liu, H. Liu, Y. Xie, Y. Yao, X. Xing, and H. Ma, “Smart image follow-up of black pigmentation on the nail with convolutional neural networks,” Available at SSRN 3834276.

[4] H. M. Sufian and G. P. Abebe, “Disease identification using finger nail image processing and ensemble nearest neighbor classifiers of color features,” Ph.D. dissertation, 2021.

[5] C. Fletcher, R. Hay, and N. Smeeton, “Observer agreement in recording the clinical signs of nail disease and the accuracy of a clinical diagnosis of fungal and non-fungal nail disease,” British Journal of Dermatology, vol. 148, no. 3, pp. 558–562, 2003.

[6] T. S. Indi and Y. A. Gunge, “Early stage disease diagnosis system using human nail image processing,” IJ Information Technology and Computer Science, vol. 7, no. 7, pp. 30–35, 2016.

[7] R. Nijhawan, R. Verma, S. Bhushan, R. Dua, A. Mittal et al., “An integrated deep learning framework approach for nail disease identification,”in 2017 13th International Conference on Signal-Image Technology &Internet-Based Systems (SITIS). IEEE, 2017, pp. 197–202.

[8] M. Yani, S. M. Budhi Irawan, S, and M. Casi Setiningsih, ST, “Application of transfer learning using convolutional neural network method for early detection of terry’s nail,” in Journal of Physics: Conference Series, vol. 1201, no. 1. IOP Publishing, 2019, p. 012052.

[9] S. L. Pinoliad, D. A. N. Dichoso, A. R. Caballero, and E. M. Albina,“Onyxray: A mobile-based nail diseases detection using custom vision machine learning,” in Proceedings of

the 5th International Conference on Information and Education Innovations, 2020, pp. 126–133.

[10] K. A. Muhaba, K. Dese, T. M. Aga, F. T. Zewdu, and G. L. Simegn, "Automatic skin disease diagnosis using deep learning from clinical image and patient information," Skin Health and Disease, vol. 2, no. 1,p. e81, 2022.

[11] A. Rehman, M. A. Khan, T. Saba, Z. Mehmood, U. Tariq, and N. Ayesha, "Microscopic brain tumor detection and classification using 3d cnn and feature selection architecture," Microscopy Research and Technique,vol. 84, no. 1, pp. 133–149, 2021.

[12] H. Hong, J. Lin, and F. Huang, "Tomato disease detection and classification by deep learning," in 2020 International Conference on Big Data,Artificial Intelligence and Internet of Things Engineering (ICBAIE).IEEE, 2020, pp. 25–29.

[13] K. Thomsen, A. L. Christensen, L. Iversen, H. B. Lomholt, and O. Winther, "Deep learning for diagnostic binary classification of multiple-lesion skin diseases," Frontiers in medicine, vol. 7, p. 574329,2020.

[14] M. Sazzadul Islam Prottasha, S. Mahjabin Farin, M. Bulbul Ahmed, M. Zihadur Rahman, A. Kabir Hossain, and M. Shamim Kaiser, "Deep learning-based skin disease detection using convolutional neural networks (cnn)," in The Fourth Industrial Revolution and Beyond: Select Proceedings of IC4IR+. Springer, 2023, pp. 551–564.

[15] K. Li, B. Ao, X. Wu, Q. Wen, E. Ul Haq, and J. Yin, "Parkinson's disease detection and classification using eeg based on deep cnn-lstm model,"Biotechnology and Genetic Engineering Reviews, pp. 1–20, 2023.

[16] P. Muthukannan et al., "Optimized convolution neural network based multiple eye disease detection," Computers in Biology and Medicine, vol. 146, p. 105648, 2022.

[17] "Reuben industriustech n dataset, "https://www.kaggle.com/datasets/reubenindustrustech /n, accessed: 26-March-2023.

[18] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Insights into imaging, vol. 9, pp. 611–629, 2018.

[19] N. Murray and F. Perronnin, "Generalized max pooling," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 2473–2480.

[20] G. Chartrand, P. M. Cheng, E. Vorontsov, M. Drozdzal, S. Turcotte, C. J.Pal, S. Kadoury, and A. Tang, "Deep learning: a primer for radiologists, "Radiographics, vol. 37, no. 7, pp. 2113–2131, 2017.