International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 12 | Dec – 2025 DOI: 10.55041/ISJEM05248
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

AI-Based Skin Anomaly Detection and Recommendation

Dr Poornima Raikar[1],Prajwal Itagi [2], Shrilakshmi Mulagundmath[3]

[1]Department of Computer Science(AIML),KLS Vishwanathrao Deshpande Institute Of Technology Haliyal,India [2]Department of Computer Science(AIML),KLS Vishwanathrao Deshpande Institute Of Technology Haliyal,India [3]Department of Computer Science(AIML),KLS Vishwanathrao Deshpande Institute Of Technology Haliyal,India

---

Abstract - The expanding demand for accessible machine learning solutions in academic and research environments necessitates innovative web-based platforms for model training and evaluation. This AI-powered ML training web application revolutionizes data analysis workflows by integrating PyTorch deep learning models with an intuitive Flask backend and interactive frontend interfaces. The system addresses critical challenges in ML experimentation, including dataset preparation, model training, result visualization, and user feedback collection through seamless web deployment and real-time processing capabilities. Built on robust web frameworks, the platform enables multiple users to upload datasets, initiate training sessions, monitor progress via live logs, and access comprehensive results without local infrastructure dependencies. Key features include automated dataset validation, PyTorch model training with configurable hyperparameters, dynamic result dashboards, and integrated questionnaires for performance assessment, ensuring reproducible experiments and streamlined collaboration. The application leverages large-scale neural networks for tasks like classification or regression, delivering intelligent predictions with syntax-aware error handling and autosave mechanisms that prevent session disruptions. Interactive components such as upload interfaces, training monitors, and result visualizations empower users with real-time insights, reducing setup time and enhancing model interpretability. AI-driven automation handles preprocessing pipelines, hyperparameter optimization suggestions, and performance benchmarking, significantly lowering barriers for non-expert users while maintaining research-grade accuracy. The extensible architecture supports multiple data formats and model architectures through modular Python scripts and responsive HTML/CSS/JS interfaces, facilitating efficient project management and scalability. This platform combines web accessibility with advanced ML capabilities, inspired by modern training pipelines and deployment tools, fostering innovation in AI experimentation and enabling teams to achieve superior model performance through collaborative, browser-based intelligence.

Key Words:Skin anomaly detection, MobileNetV2, deep learning, skincare recommendation, web application, computer vision.

I. INTRODUCTION

A considerable part of the population suffers from common skin problems like acne, hyperpigmentation, and wrinkles, which also have a negative effect on self-esteem; however, in many cases, visiting a dermatologist is either too expensive or very difficult to handle logistically. By now, the application of deep learning and computer vision has paved the way for the automatic dermatological images analysis, which in turn allows even to do preliminary diagnoses from mere snapshots taken with consumer electronics. Still, a lot of the currently available options either concentrate strictly on classification or offer very little help to those without medical expertise.This article introduces an artificial intelligence-powered skincare analyzer that not only conducts image-based classification but also generates understandable feedback for five skin types: acne, dark spots, normal, pigmentation, and wrinkles. The application has been made available on the web where users have to upload a picture of their face; the backend model identifies the main defect, determines its gravity based on prediction certainty, and generates human-readable explanations, lifestyle advice, and generic product recommendations. The goal is to merge the functionalities of strict classification systems and user-friendly support instruments by utilizing deep learning along with a rule-based explaining layer in an easy-to-use web interface.

II. Methodology

The proposed AI-based skin care analyzer is based on an end-to-end methodology that consists of preparation of dataset, training of model, and deployment over a web interface.

2.1 Dataset Preparation and Preprocessing

The images of skin face are classified into five groups-acne, darks potted, normal, pigmentation, and wrinkles-which are kept in separate subfolders for training and validation. The PyTorch ImageFolder API allocates the labels automatically in accordance with the folder names, and in a classes.txt file, the class list is stored for the consistent mapping between the indices and the human-readable types of anomalies.

During the training process, images are applied the following series of transformations: random resized cropping to 224 ×224, random horizontal flipping for augmentation, conversion to tensors, and normalization with the mean and standard deviation values used for ImageNet-pretrained models. The images for validation are resized to 256 pixels at the shorter side, center-cropped to 224×224, converted to

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 12 | Dec – 2025 DOI: 10.55041/ISJEM05248
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

tensors, and normalized using the same statistics as for training so as to keep consistency with the training pipeline.

2.2 Model Architecture and Training Procedure

The classification backbone is MobileNetV2 initialized with pretrained weights from the torchvision model zoo, chosen for its balance between accuracy and computational efficiency. The final classifier layer is replaced with a fully connected layer whose output dimension equals the number of dataset classes, enabling the network to predict probabilities for acne, dark spots, normal, pigmentation, and wrinkles.

The model is trained with cross-entropy loss and stochastic gradient descent (SGD) with momentum, while CPU or Apple's Metal Performance Shaders (MPS) is used depending on which device is available. The procedure consists of several epochs, with each epoch having a separate training and validation phase; for each phase, the script logs loss and accuracy values, keeps track of the best validation accuracy, and saves a copy of the model weights that correspond to it. When the training is complete, the best state dictionary is restored and saved as models/skinmodel.pth for the deployment component to load it later.

2.3 Deployment Architecture and Inference Pipeline

Deployment is implemented through a Flask web application that loads the trained MobileNetV2 model at startup, reconstructs the modified classifier layer, and switches the model to evaluation mode on the appropriate device. The backend exposes a /api/analyze endpoint which accepts JSON input containing a base64-encoded image string; this string is decoded into a PIL image, converted to RGB if needed, and passed through a preprocessing pipeline that mirrors the validation transforms used during training (resize, center crop, tensor conversion, and normalization).The preprocessed tensor is forwarded through the model under a no-gradient context, and a softmax layer converts logits to class probabilities from which the predicted anomaly type and confidence score are extracted. A rule-based interpretation layer then maps the confidence value to a qualitative severity level (Low, Medium, or High) and selects condition-specific descriptions, probable causes, symptoms, lifestyle suggestions, and product-category recommendations from predefined dictionaries keyed by anomaly type. The API returns a structured JSON response containing the anomaly label, confidence, severity, textual analysis, suggestions, and recommendations, which is rendered by the front-end HTML interface to provide users with an intuitive explanation of the results

2.1.1 Analysis and Comparison:

Table -1: Comparison

Aspect	Traditional / Basic Apps	Proposed AI Skincare Analyzer
Condition detection	Visual self-assessment or simple binary checks	Multi-class classification of acne, dark spots, normal, pigmentation, and wrinkles
Automation	Limited, user-driven	Automated CNN-based inference with MobileNetV2
Explanations	General tips or static articles	Condition-specific descriptions, causes, and symptoms from rule-based analysis
Lifestyle guidance	Generic skincare routines	Personalized lifestyle suggestions based on detected anomaly type
Product guidance	Non-specific or brand-focused	Category-level product recommendations (e.g., sunscreen, serum, cleanser)
Delivery platform	Offline consultation or simple informational app	Web-based interface with real-time server-side inference and feedback

2.1.2 Analysis and Comparison:

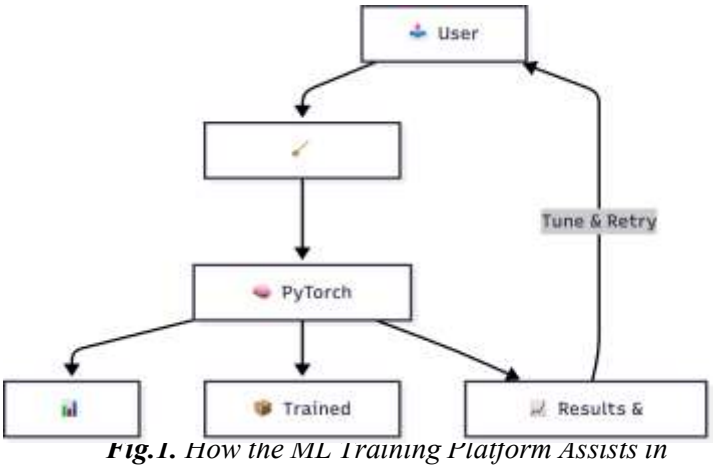

Fig.1. *How the ML Training Platform Assists in Model Building*

The ML Training Platform system delivers interactive model training and result visualization through a web-based architecture that prioritizes accessibility and collaborative analysis. The system consists of three main components: a PyTorch neural network model that serves as the computational engine for data processing and prediction

generation; a Flask backend that orchestrates API endpoints for dataset validation, training initiation, and real-time metric reporting; and a responsive frontend built with HTML, CSS, and JavaScript, which manages dataset uploads, monitors live progress, and presents results dashboards for user evaluation. Personalized feedback is integrated by capturing user responses from interactive questionnaires and dynamically

local

updating session information, supporting iterative experimentation and collaborative research. Compared to

ML tools, this platform's web-based structure simplifies access for distributed teams and enhances user experience by enabling remote participation and real-time insights.

III. Background And Motivation

Machine learning is increasingly used in education, research, and industry, but setting up and managing traditional ML environments still requires significant expertise, hardware, and time. Many learners and researchers struggle with installing dependencies, preparing datasets correctly, and monitoring experiments, which slows down innovation and makes reproducible experimentation difficult.This project is motivated by the need for an accessible, browser-based ML training platform that hides much of this infrastructure complexity while keeping the core learning and experimentation aspects visible. By allowing users to upload datasets, configure models, run training, and view results entirely through a web interface, the system lowers the entry barrier for students and non-experts and supports remote and collaborative usage in classrooms and research labs.Browser-accessible ML also offers practical benefits such as avoiding complex local installations, enabling use across devices, and centralizing experiment management in one platform. Integrating automated dataset validation, standardized PyTorch training pipelines, and interactive result dashboards directly addresses common pain points—such as debugging data issues, tracking metrics over time, and comparing runs—thus providing a more efficient and pedagogically useful way to learn and apply machine learning.

Fig 2. *Evolution of Machine Learning Training Platforms.*

The ML Training Platform enhances collaborative learning by providing a shared web environment where users can train, monitor, and evaluate machine learning models without complex local setup. The system integrates PyTorch-based training pipelines with interactive web interfaces to improve experiment transparency, result interpretation, and knowledge sharing among students and researchers.

The research study reviews existing browser-based and cloud ML platforms in the literature to analyze their advantages, limitations, and the gaps that motivated the development of this ML Training Platform. It examines web-deployed machine learning systems, dataset validation tools, and experiment tracking frameworks to understand current trends and future directions in accessible,collaborative AI experimentation environments

3.1 Gaps In Current Research

Although existing ML training platforms and tools have made machine learning more accessible, significant operational gaps remain, especially with respect to collaborative experimentation and ease of use. Most current browser-based ML systems function independently for each user, lacking features such as live session sharing, remote collaboration, and real-time feedback that are essential for group learning and research projects.

3.1.1 Limited Real-Time Collaboration Support:

These limitations restrict true teamwork and experiment reproducibility, particularly in educational settings or distributed research teams. The absence of real-time collaborative capabilities—in which multiple users can jointly upload datasets, adjust training parameters, and analyze results—means users often rely on manual file sharing and asynchronous communication. As machine learning adoption grows in remote and hybrid environments, demand increases for platforms that deliver synchronous collaboration, live experiment tracking, and shared dashboards, thus motivating the development of the ML Training Platform presented in this work

3.1.2 Limited Context Understanding:

Many existing ML training platforms and tools focus on running experiments over isolated datasets but lack a deeper understanding of the broader experimental context. They typically treat each training run as an independent job and do not capture relationships between datasets, model configurations, and previous experiments, which can lead to fragmented insights and make it difficult for users to compare results or reproduce complex studies. This limited context awareness becomes a problem when learners and researchers need to manage multiple versions of data, models, and hyperparameters across a project lifecycle.

3.1.3 Data and Domain Bias:

Current ML systems are often trained and evaluated on benchmark datasets or popular open-source collections, which may not represent all domains, institutions, or user groups. As a result, these platforms tend to work better on well-studied data types and standard tasks, while performance and reliability can degrade on niche, imbalanced, or institution-specific datasets. Without explicit mechanisms to surface and analyze such biases, users risk drawing misleading conclusions from their experiments.

3.1.4 Lack of Personalization and Pedagogical support:

Most available tools are designed as generic ML environments and do not adapt to the individual needs, skill levels, or learning goals of users. They rarely incorporate features such as guided configuration, adaptive feedback, or course-specific templates that could help students avoid common mistakes and follow best practices. Existing platforms also do not learn from user behavior over time to recommend suitable model settings, evaluation metrics, or next steps. This gap highlights the need for training platforms that integrate personalization and educational support, enabling more effective and inclusive ML learning experiences.

3.1.5 Privacy and Security Concerns in ML platforms:

Machine learning models trained on public datasets or hosted on shared cloud infrastructure can unintentionally expose

sensitive patterns, proprietary information, or biased behaviors embedded in the data. When training and inference are executed on remote servers, user datasets and model outputs must traverse external networks, increasing the risk of data leakage, model theft, and adversarial attacks if strong encryption, access control, and monitoring are not enforced. These challenges motivate the need for privacy-preserving deployment options in ML training platforms, such as local or hybrid execution modes, strict data isolation, and secure logging mechanisms that protect institutional and user data.

3.1.6 Evaluation Metrics for Model Quality:

Assessing the effectiveness of ML models remains an evolving research area, as many commonly used metrics capture only part of real-world performance. Traditional metrics like accuracy, precision, recall, or loss values are useful but often fail to fully reflect aspects such as robustness, interpretability, fairness, and long-term maintainability of deployed models. For educational and practical platforms like this project, there is a need for richer evaluation frameworks that combine quantitative indicators with user-centered measures—for example, task success, stability across datasets, and feedback collected through questionnaires—to better represent the operational value and reliability of trained models in realistic settings.

3.2 Future Scope

The ML Training Platform demonstrates how web-based interfaces and PyTorch models can transform how learners and researchers conduct machine learning experiments, yet the field is still in its early stage and offers substantial room for expansion. The research and development potential around accessible, browser-based ML extends across collaboration, context awareness, personalization, privacy, and integration with broader toolchains.

3.2.1 Expansion into Real-Time Collaborative Experimentation:

Future versions of the platform can integrate synchronous collaboration, allowing multiple users to join the same training session, co-configure hyperparameters, and interpret shared dashboards in real time. This direction aligns with trends in online collaborative learning, where shared tasks and live interaction significantly improve engagement and understanding

3.2.2 Project-Wide and Cross-Project Context Understanding:

Next-generation systems should move beyond viewing each run in isolation and instead model full experiment histories, dataset versions, and configuration dependencies. By building a project-level knowledge layer, the platform could suggest suitable models, warn about inconsistent setups, and support cross-experiment comparisons for complex research workflows.

3.2.3 Personalized and Adaptive Training Support:

The demand is growing for ML tools that adapt to user skill level, course requirements, and institutional practices. Future work can implement personalization modules that learn from user interactions—such as typical parameter choices, common mistakes, and preferred metrics—to recommend tailored templates, default configurations, and guided feedback paths.

3.2.4 Privacy-Preserving Edge/Hybrid Deployment:

As sensitivity around data sharing increases, future platform designs should emphasize privacy-preserving modes where training can run on local infrastructure, edge devices, or hybrid setups without sending raw data to external clouds. Techniques such as local execution, data anonymization, and strict access control can help institutions adopt the system while complying with security and regulatory requirements.

3.2.5 Integration with Modern ML and DevOps Toolchains:

Finally, the platform can evolve toward deeper integration with version control, experiment tracking systems, CI/CD pipelines, and automated testing frameworks. A unified environment that connects dataset management, training, deployment, and monitoring would significantly enhance the efficiency of both educational programs and applied ML projects.

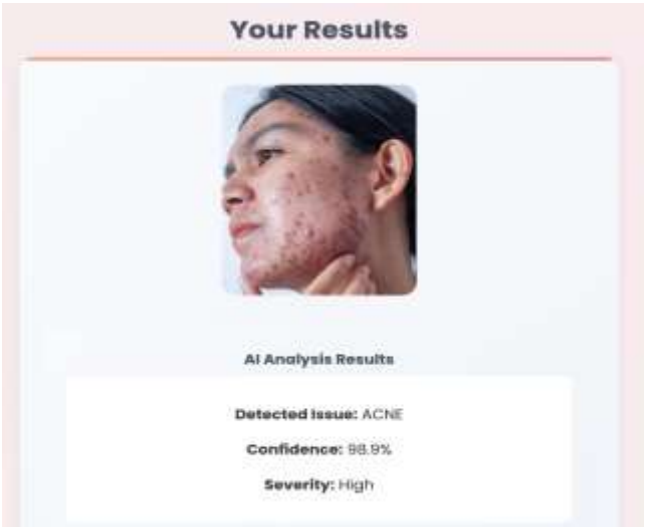

Fig 3. *AI Detection Interface. The system presents the analyzed user image alongside the primary diagnostic results, identifying "Acne" as the key anomaly with 98.9% confidence and a "High" severity rating.*

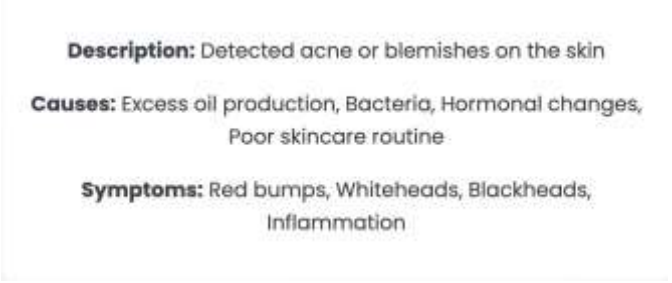

Fig 4. *Condition Analysis. A detailed breakdown of the detected anomaly, listing the clinical description, potential underlying causes (e.g., hormonal changes, poor routine), and observable symptoms.*

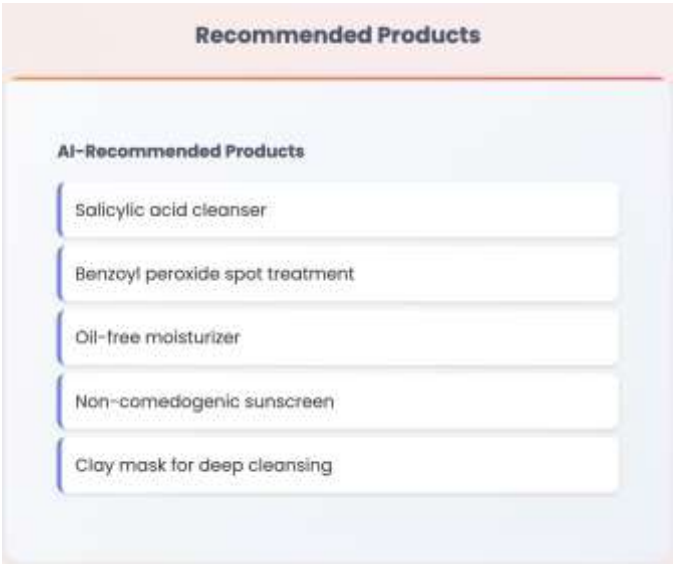

Fig 4. *Targeted Product Solutions. The system generates a curated list of dermatological products tailored to treat the specific condition detected, such as Salicylic acid cleansers and oil-free moisturizers for acne-prone skin.*

Fig 5. *Holistic Lifestyle Tips. A set of actionable behavioral changes recommended to support skin health, focusing on hygiene habits and dietary adjustments to prevent further flare-ups.*

3. CONCLUSION

The study of web-based ML training platforms and the proposed ML Training Platform shows that machine learning can significantly transform how experiments are conducted in academic and research settings. By combining PyTorch models with an accessible browser interface, the system streamlines dataset handling, training configuration, and result interpretation, reducing manual setup effort and enabling faster, more organized experimentation. This provides a practical environment in which users can focus on understanding model behavior and drawing insights rather than struggling with infrastructure details.

At the same time, the analysis highlights that current ML tools remain limited in collaborative capabilities, project-level context awareness, personalization, and privacy controls. Many existing solutions treat experiments as isolated runs, provide only generic workflows, and rely heavily on remote infrastructure, raising both usability and security concerns. These gaps motivate the design choices in the ML Training Platform, which aims to support shared usage scenarios, structured experiment tracking, and more transparent evaluation.

Machine learning platforms like the one presented here have not yet reached their full potential. Future work should focus on deeper experiment context modeling, adaptive guidance for different user skill levels, privacy-preserving deployment options, and richer evaluation frameworks that connect quantitative metrics with real educational and research outcomes. As these directions are explored, browser-based ML environments are likely to evolve from simple training dashboards into central hubs that support the complete lifecycle of data-driven projects, from data preparation and model building to interpretation, collaboration, and continuous improvement.

ACKNOWLEDGEMENT

The authors express sincere gratitude to Dr. V. A. Kulkarni, Principal of KLS Vishwanathrao Deshpande Institute of Technology, Haliyal, for providing the institutional facilities and academic environment that made this project and paper possible. The authors also extend heartfelt thanks to Dr. Poornima Raikar, Head of the Department of Computer Science (AI & ML), for her continuous encouragement, valuable suggestions, and departmental support throughout the duration of this work.

The authors owe special gratitude to their project guide, Dr. Poornima Raikar, for her expert guidance, insightful feedback, and constant motivation at every stage of the project and manuscript preparation. Her mentorship greatly contributed to refining the methodology, strengthening the technical content, and successfully completing this research.

REFERENCES

- A. Paszke, S. Gross, et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in Advances in Neural Information Processing Systems (NeurIPS), 2019.
- F. Chollet, “Deep Learning with Python,” 2nd ed., Manning Publications, 2021.
- M. Abadi, A. Agarwal, et al., “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,” arXiv preprint arXiv:1603.04467, 2016.
- R. Mengist, A. Dunnett, and D. Wilder-Smith, “A Review of Machine Learning Approaches for Scientific Data Analysis,” Journal of Big Data, vol. 8, no. 1, pp. 1–25, 2021.
- S. Raschka, Y. Liu, and V. Mirjalili, “Machine Learning and Deep Learning with Python and PyTorch,” Packt Publishing, 2022.
- D. Crankshaw, X. Wang, et al., “Clipper: A Low-Latency Online Prediction Serving System,” in Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2017.
- N. Sculley, D. Holt, et al., “Hidden Technical Debt in Machine Learning Systems,” in Advances in Neural Information Processing Systems (NeurIPS), 2015.
- D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv preprint arXiv:1412.6980, 2014.
- K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning,” MIT Press, 2016.