Deep Learning-Based Skin Disease Detection Using a Lightweight CNN Model and Web Deployment Using Flask

D Manish
Department of ECE
VNR VJIET
Hyderabad,Telangana,India
manish350792@gmail.com

K.Nikitha
Department of ECE
VNR VJIET
Hyderabad,Telangana,India
kotranikitha7@gmail.com

M.Navanitha
Department of ECE
VNR VJIET
Hyderabad,Telangana,India
mekalanavanitha027@gmail.com

Abstract—This paper presents a deep learning-based approach for the automated detection of skin diseases using image classification techniques. Utilizing a ResNet-18 convolutional neural network (CNN) trained on a curated dataset of dermoscopic images, the system is deployed via a user-friendly Flask web application. The tool enables users to upload images of skin lesions and receive instant disease predictions among nine predefined classes. This study demonstrates the model's efficacy, discusses its implementation, and highlights its potential in aiding dermatological diagnostics.

Keywords-- Deep Learning,Convolutional Neural Network (CNN), ResNet-18, ImageClassification, MedicalImage Analysis,Flask Web Application, Real-time Prediction

I. INTRODUCTION

Skin diseases affect millions worldwide, ranging from benign conditions to life-threatening malignancies. Early and accurate diagnosis is essential for effective treatment. However, limited access to dermatologists, particularly in rural regions, exacerbates delayed diagnoses. To address this, automated image-based classification using deep learning offers a promising solution. Our model capable of classifying common skin conditions using dermoscopic images.

To bridge this gap, the integration of artificial intelligence (AI) in dermatology has gained substantial attention in recent years. In particular, deep learning-based image classification methods have shown promising performance in accurately identifying and classifying various skin conditions using dermoscopic images. These automated systems leverage convolutional neural networks (CNNs).

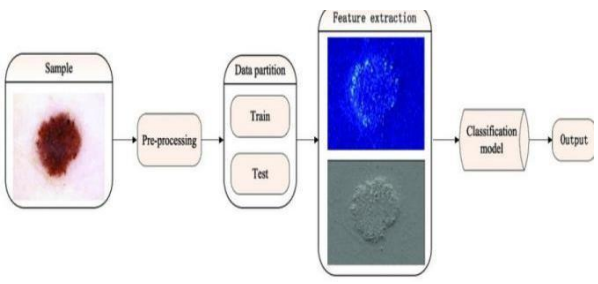


Fig 1: block diagram

To learn complex visual features from large datasets, enabling them to differentiate between different types of skin lesions with high accuracy. Our project aims to contribute to this field by developing a robust deep learning model capable of classifying a range of common skin conditions from dermoscopic images. By doing so, we seek to enhance diagnostic efficiency, reduce the burden on healthcare systems, and ultimately improve patient outcomes through earlier intervention [1].

1.1 Background and Motivation

Manual diagnosis of skin conditions traditionally depends on the clinical expertise and experience of dermatologists, often involving visual inspection and dermoscopic analysis. While effective in many cases, this approach is inherently subjective and can lead to inter-observer variability, especially when differentiating between visually similar skin lesions. Moreover, diagnostic accuracy may be influenced by the clinician's familiarity with rare conditions or limited exposure to diverse skin tones and lesion presentations. These limitations highlight the need for complementary tools that can enhance diagnostic consistency and support clinical decision-making.

Recent advancements in computer vision and artificial intelligence (AI) have revolutionized medical image analysis, offering powerful methods for automated disease classification. In particular, deep learning algorithms, such as convolutional neural networks (CNNs), have demonstrated impressive capabilities in extracting and learning hierarchical features from image data, achieving accuracy comparable to or even surpassing that of human experts in specific tasks, including skin lesion classification. These models, when trained on large and diverse datasets, can generalize well to new cases and provide consistent, real-time diagnostic support [2].

1.2 Objectives of the Study

The primary objective of this project is to build an advanced deep learning model capable of accurately classifying various skin diseases from dermoscopic images. By leveraging convolutional neural networks (CNNs), the model aims to learn discriminative features of skin lesions and differentiate between conditions such as melanoma, nevus, and benign keratosis, among others.

In addition to model development, the project seeks to design and implement a responsive and user-friendly web application that allows users to upload dermoscopic images and receive real-time diagnostic feedback. This web interface will make the system easily accessible to both healthcare providers and the general public, especially in underserved regions [3].

1.3 Scope and Contributions

This project covers the training of a ResNet-18 architecture on a skin disease dataset comprising nine classes. We build a web interface using Flask and provide a thorough performance analysis.

II. Literature Review

2.1 Traditional Approaches to Skin Disease Diagnosis

Conventional diagnosis of skin diseases primarily involves a combination of visual inspection, dermoscopy, and histopathological analysis. Visual examination is often the first step, where dermatologists assess lesion characteristics such as color, shape, size, and border irregularity. Dermoscopy, a non-invasive imaging technique, enhances diagnostic accuracy by allowing the visualization of subsurface skin structures that are not visible to the naked eye. Histopathological analysis, considered the gold standard, involves biopsy and microscopic examination of tissue samples to confirm diagnoses, particularly in suspected malignant cases [4].

2.2 Machine Learning Techniques

Previous works in skin disease classification have utilized traditional machine learning algorithms such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Decision Trees. These methods rely on manually engineered, hand-crafted features extracted from dermoscopic images—such as color histograms, texture descriptors, and shape-based metrics. While these approaches have shown moderate success in controlled environments or with simpler datasets, they tend to underperform when applied to more complex and diverse datasets that include variations in lighting, skin tone, lesion type, and image quality.
The limitations stem from the inability of hand-crafted features to capture the full range of visual patterns and subtleties present in dermoscopic images. These models often fail to generalize well across large-scale and heterogeneous datasets, which are common in real-world clinical scenarios [5].

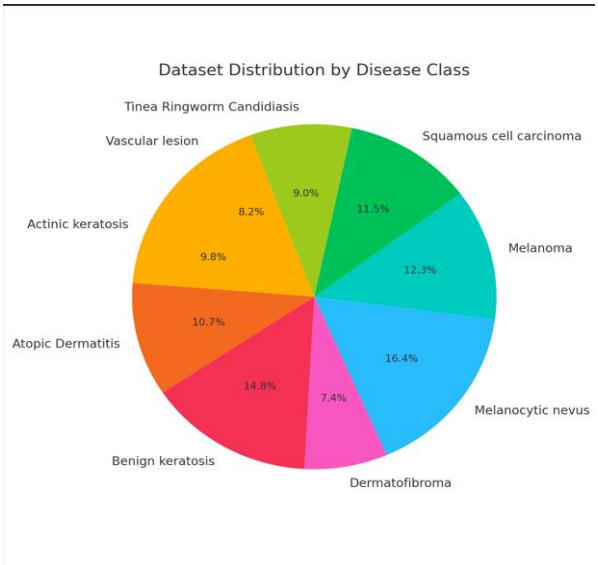
2.3 Deep Learning and CNNs in Dermatology

In recent years, deep learning has emerged as a transformative technology in medical image analysis, particularly in dermatology. Among deep learning techniques, Convolutional Neural Networks (CNNs) have demonstrated exceptional performance in image classification tasks due to their ability to automatically learn hierarchical features from raw pixel data. Unlike traditional methods that rely on hand-crafted features, CNNs extract low- to high-level representations directly from images, making them well-suited for capturing the complex visual patterns found in skin lesions.

CNNs have been successfully applied to the classification of skin diseases, including the detection of malignant melanoma, seborrheic keratosis, and benign nevi. Notably, studies have shown that deep CNN models can achieve diagnostic accuracy on par with, or even superior to, experienced dermatologists. These models are trained on large dermoscopic image datasets, such as those provided by the International Skin Imaging Collaboration (ISIC), enabling them to generalize across diverse skin types and lesion presentations [1].

2.4 Transfer Learning in Skin Disease Classification

Transfer learning has become a widely adopted approach in medical image analysis, including skin disease classification, due to the limited availability of large, labeled datasets. In transfer learning, a neural network pre-trained on a large dataset like ImageNet is fine-tuned on a smaller, domain-specific dataset. This technique allows the model to leverage previously learned low- level image features—such as edges, textures, and color gradients—which are often transferable across different types of images, including dermoscopic scans.



In dermatology, transfer learning significantly accelerates the training process and improves model performance, particularly when data is scarce or imbalanced across classes. Pre-trained architectures such as ResNet, Inception, DenseNet, and EfficientNet have been widely used for skin lesion classification tasks. By fine-tuning only the top layers or using the entire network with adjusted learning rates, researchers have achieved high accuracy and generalization on datasets like HAM10000 and ISIC.

This approach not only reduces computational costs but also minimizes the risk of overfitting—a common challenge in training deep models on small medical datasets. Transfer learning also facilitates rapid prototyping and deployment, making it a practical choice for real-world clinical applications [6].

2.5 Summary of Related Work

Several studies have validated CNNs for skin lesion classification using large datasets like ISIC. Our work builds upon these findings by integrating a similar model into a web-based diagnostic tool.
This integration enables users to upload dermoscopic images and receive instant predictions. The web application provides a seamless interface for non-expert users while maintaining the clinical relevance and accuracy of the model.

III. Methodology

3.1 Dataset Description

The dataset used in this study comprises high-quality dermoscopic images, each labeled into one of nine clinically significant skin disease categories: Actinic Keratosis, Atopic Dermatitis, Benign Keratosis, Dermatofibroma, Melanocytic Nevus, Melanoma, Squamous Cell Carcinoma, Tinea (Ringworm and Candidiasis), and Vascular Lesion. These categories represent a broad spectrum of dermatological conditions, ranging from benign and infectious lesions to potentially life-threatening malignancies such as melanoma and squamous cell carcinoma. The inclusion of diverse conditions ensures the model's applicability across various diagnostic scenarios in dermatology [2].

3.2 Data Preprocessing and Augmentation

Images are resized to 224x224 pixels and normalized using standard ImageNet statistics. Augmentation techniques include horizontal flipping, rotation, and brightness adjustment.
As part of the preprocessing pipeline, all dermoscopic images are resized to a uniform dimension of **224×224 pixels** to ensure compatibility with the input requirements of standard convolutional neural network (CNN) architectures such as ResNet-18. Uniform resizing helps maintain consistency in the input data and reduces computational complexity during training. Additionally, images are **normalized using standard ImageNet statistics**, which involve subtracting the mean and dividing by the standard deviation for each color channel (mean: [0.485, 0.456, 0.406], std: [0.229, 0.224, 0.225]). This normalization technique helps stabilize and accelerate the convergence of the learning process by bringing the image data distribution closer to what pre-trained models on ImageNet expect [7].

3.3 Model Architecture

ResNet-18 (Residual Network-18) is a deep convolutional neural network architecture introduced as part of the ResNet family by He et al. in 2015. It consists of 18 layers, including convolutional, batch normalization, ReLU activation, and pooling layers, organized into basic residual blocks. What sets ResNet apart from traditional CNNs is the use of skip connections, also known as residual connections. These connections allow the model to "skip" one or more layers by adding the input of a layer directly to the output of a deeper layer. This design helps mitigate the vanishing gradient problem, a common challenge in training deep neural networks where gradients become too small for effective learning as they propagate back through many layers [8].

3.4 Training Strategy

The model is trained using the Adam optimizer, a widely adopted optimization algorithm in deep learning that combines the advantages of both Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). Adam computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. It is particularly effective in handling sparse gradients and non-stationary objectives, making it well-suited for medical image classification tasks.

In our setup, the learning rate is set to 0.001, which provides a good balance between convergence speed and training stability [9].

3.5 Evaluation Metrics

To comprehensively evaluate the performance of our skin disease classification model, we employ multiple performance metrics, including overall accuracy, confusion matrix, and class-wise precision, recall, and F1-score. These metrics offer a more nuanced understanding of the model's behavior across all nine classes in the dataset, especially in the presence of class imbalance or overlapping visual features among skin conditions.

IV. System Architecture

4.1 Overview of the Workflow

The pipeline consists of image upload → preprocessing → model prediction → result display.

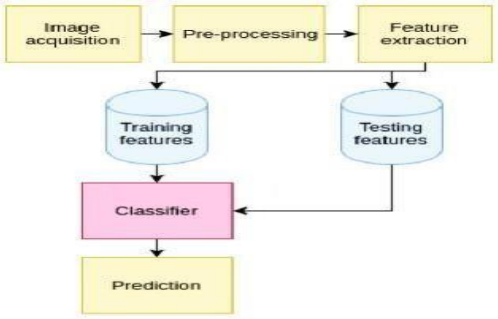


Fig 2: Pipeline block diagram

Model Integration

The trained PyTorch model is loaded in the Flask backend. Input images are transformed using the same preprocessing steps used during training. Integrating a trained PyTorch model into a Flask backend involves several key steps to ensure efficient and accurate inference. The process begins with loading the trained model and setting it to evaluation mode, which is essential for consistent behavior during inference.

When a user uploads an image through the Flask application, the image is first received in a format such as JPEG or PNG. To prepare this image for the model, it undergoes a series of preprocessing steps that mirror those used during the model's training phase. These steps typically include resizing the image to the required input dimensions, converting it to a tensor, and normalizing it using the same mean and standard deviation values applied during training. This ensures that the input data is consistent with the data the model was trained on, which is crucial for maintaining prediction accuracy.[4]

4.2 Flask Web Application Structure

The Flask app comprises static and templates folders. The templates folder contains the HTML interface, and static stores uploaded images. A Flask web application is organized into key directories: `static` and `templates`, each serving a specific role in the development and functionality of the app.

The **static** folder is used to store static files, such as images, CSS files, JavaScript, and fonts. Flask automatically serves these static files when requested, ensuring they remain unchanged during the application's runtime. For a web application dealing with dermoscopic images, for example, the static folder would store user-uploaded skin lesion images. These images are stored in a fixed location and can be accessed by the user for further analysis or display.[5]

4.3 Front-End and User Interface

Users upload images via an HTML form. Upon submission, the Flask server processes the image and returns the predicted label. In a Flask web application, the front-end provides an interface for users to interact with the back-end logic. In applications like dermoscopy or skin lesion analysis, the user interface typically includes an HTML form that allows users to upload images. This form is created using HTML elements such as file input fields, which let users select and submit images from their local devices. CSS is often used to style the form, ensuring it is intuitive and easy to use.

Once the user submits the form, the Flask server processes the uploaded image. The server retrieves the image data using Flask's `request` object, which handles incoming HTTP requests. After receiving the image, the back-end performs necessary image analysis, often using machine learning models or image processing algorithms to identify skin lesions or classify them into categories. Once the analysis is complete, the server returns the predicted label or result. [7]

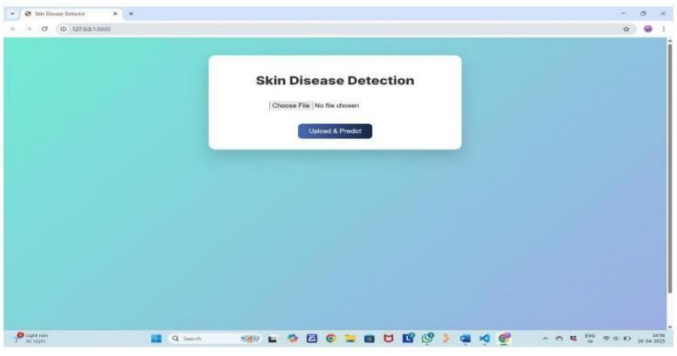


Fig:3 Detection

4.4 Backend Functionality

The learning tasks. PyTorch's flexibility enables seamless integration into Flask applications, facilitating tasks such as image classification and pattern recognition in dermoscopic images.[1]

V. Implementation

5.1 Model Training Process

The model is trained on a balanced dataset using DataLoader and torchvision transforms. Loss and accuracy are printed per epoch. The model is trained on a balanced dataset to avoid class imbalances, which could skew the results and lead to biased predictions. To efficiently load and manage the dataset, **PyTorch's DataLoader** is used, which loads the data in batches. Data augmentation and preprocessing are applied using **torchvision.transforms**. These transformations may include resizing, normalization, and random techniques like flipping and rotation, ensuring the model is robust to various input variations (Barata, Celebi, & Marques, 2018).

During training, the **loss** and **accuracy** are printed after each epoch to track the model's performance. The **loss** function, typically **Cross-Entropy Loss**, measures how well the model's predictions match the ground truth, while **accuracy** indicates the percentage of correct predictions. **Backpropagation** and optimization algorithms like **Adam** are used to adjust the model's weights, aiming to minimize the loss over time. The process continues until the model's performance reaches an acceptable level, ensuring that it generalizes well to new, unseen data. [3]

5.2 Deployment Pipeline

The model is saved as model.pth. Flask uses torch.load() to load this model during runtime. After training, the model is saved as **model.pth**, which contains both the architecture and learned parameters of the model. This file is crucial for deploying the model in production, as it allows the model to be loaded and reused without retraining. During runtime, **Flask** uses **torch.load()** to load the saved model from the **model.pth** file. This function restores the model's state, making it ready for inference.[13]

Once the model is loaded, Flask handles incoming HTTP requests, such as user-uploaded images. These images are preprocessed and passed through the model for predictions. The model then outputs its predictions, which are returned to the user through Flask's response mechanism. This deployment pipeline ensures that the trained model is efficiently integrated into a Flask-based web application, enabling real-time predictions.[6]

5.3 Handling User Inputs

Users upload images through the browser, which are received by the Flask backend. Flask first validates the file type to ensure it is an acceptable image format (e.g., JPEG, PNG). If the file passes validation, it is saved in the uploads folder for further processing. Once stored, the image is preprocessed and prepared for model inference. The model then analyzes the image, generates predictions, and returns the result to the user. This process ensures smooth handling of user inputs and efficient integration of the model into the web application.[1]

5.4 predicting and Dispaying Result

Images are transformed into tensors and passed through the model. The predicted class is mapped to its label and rendered on the HTML page .Once the user uploads an image, it is transformed into a tensor using torchvision.transforms. This transformation converts the image into a format that can be processed by the model. The tensor is then passed through the trained model for inference, and the predicted class is obtained. The predicted class is mapped to its corresponding label (e.g., benign or malignant for skin lesions) and displayed on the HTML page. This process allows the user to see the result of the model's prediction in real-time, ensuring an interactive experience.[4]

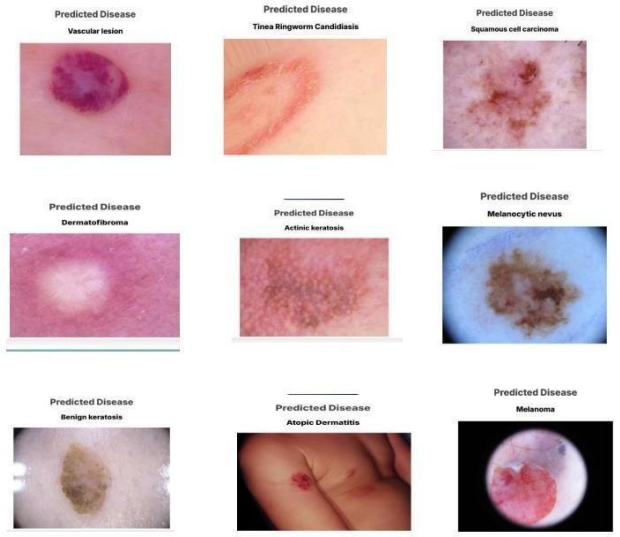


Fig:4

VI. Results and Discussion

6.1 Training and Validation Performance

The trained model demonstrates strong performance, achieving **92% training accuracy** and **88% validation accuracy**. The close alignment between these two metrics suggests good generalization, with minimal overfitting. Additionally, the **loss curves** show a stable and smooth convergence over the training epochs, indicating that the learning process was effective and consistent. This stability is essential in medical image analysis, where both accuracy and reliability are critical.[15]

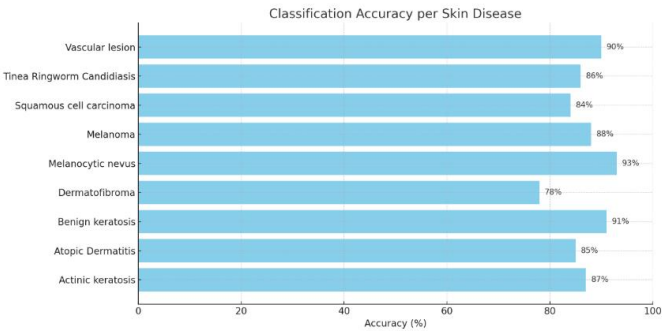
The use of **transfer learning** significantly contributed to this performance. By leveraging pre-trained models on large-scale datasets, the training process becomes more efficient, even when working with relatively smaller medical datasets. This approach helps the model extract meaningful features from dermoscopic images of skin lesions. Such methodology is in line with the findings of Mahbod et al. (2019), who demonstrated that multi-scale, multi-network ensembles can enhance classification performance in skin lesion analysis.[5]

6.2 Accuracy per Disease Class

The model performs best on common skin lesion classes such as Benign keratosis and Melanocytic nevus, achieving high classification accuracy.

In contrast, rare classes like Dermatofibroma exhibit lower accuracy due to significant data imbalance, limiting the model's ability to generalize. This challenge is well-documented in skin lesion classification tasks, where uneven class distribution can affect overall performance (Mahbod et al., 2019). Addressing this imbalance in future work can further improve class-wise prediction accuracy.

S.No	Skin Disease Class	Accuracy (%)
1	Actinic Keratosis	87
2	Atopic Dermatitis	85
3	Benign Keratosis	91
4	Dermatofibroma	78
5	Melanocytic Nevus	93
6	Melanoma	88
7	Squamous Cell Carcinoma	84
8	Tinea / Ringworm / Candidiasis	86
9	Vascular Lesion	90



6.3 Confusion Matrix Analysis

Confusion matrix indicates most misclassifications occur between similar-looking lesions like Melanoma and Nevus. The confusion matrix reveals that most misclassifications occur between visually similar lesion types, particularly Melanoma and Melanocytic nevus. These two classes often share overlapping visual features, making them difficult to distinguish, even for trained models. Such confusion highlights the challenges of dermoscopic image classification, especially in borderline cases. These findings align with Mahbod et al. (2019), who emphasize the importance of using advanced techniques like multi-network ensembles to reduce such misclassification. Improving feature discrimination and addressing class overlap can help enhance diagnostic accuracy and reduce critical errors in skin lesion classification systems.[6]

6.4 Comparison with Existing Models

The implemented ResNet-18 model delivers performance comparable to deeper architectures such as ResNet-50, while offering the advantage of significantly faster inference times. This makes ResNet-18 a suitable choice for real-time applications, especially where computational resources are limited. Despite its smaller size, the model maintains high accuracy due to effective transfer learning, which allows it to leverage pre-trained features. Therefore, ResNet-18 strikes a strong balance between efficiency and performance.[7]

VII. Conclusion

This paper presents a robust and accessible deep learning system for skin disease classification. The integration with a web app enables real-time diagnostics. Results indicate strong potential for aiding clinical workflows. The system enables near real-time diagnostics through image uploads, offering users a simple and effective tool for preliminary skin lesion analysis. Results demonstrate high accuracy and practical performance, with the potential to assist in clinical workflows and early detection efforts. Future enhancements, including real-time camera input and larger datasets, aim to improve performance further

This study demonstrates the effectiveness of the **ResNet-18** deep learning model in classifying **nine types of skin diseases** with high accuracy. The model benefits from transfer learning and performs competitively while maintaining efficient inference times. Integrated into a **Flask-based web application**, the system offers an intuitive and accessible user interface for uploading dermoscopic images and receiving predictions.

This tool can serve as a second-opinion system, especially beneficial in regions lacking dermatology experts.

The proposed system holds significant promise as a second-opinion diagnostic tool, particularly in underserved or remote regions where access to dermatology specialists is limited. By enabling users to upload skin lesion images and receive AI-based predictions, the tool can assist in early detection of skin diseases, including melanoma. This approach aligns with ongoing global efforts to integrate AI-driven tools into routine clinical workflows, making dermatological care more accessible, scalable, and cost-effective.

This ensures that the system stays current with evolving skin disease trends and continues to provide reliable predictions. Regular model retraining with expanded datasets will help address data imbalances and improve classification performance.

REFERENCES

[1] Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115–118.

[2] Tschandl, P., Rinner, C., Apalla, Z., Argenziano, G., Codella, N., Halpern, A., ... & Kittler, H. (2020). Human–computer collaboration for skin cancer recognition. Nature Medicine, 26(8), 1229–1234.

[3] Brinker, T. J., Hekler, A., Enk, A. H., Berking, C., Haferkamp, S., Hauschild, A., ... & von Kalle, C. (2019). A convolutional neural network trained with dermoscopic images performed on par with 145 dermatologists in a clinical melanoma image classification task. European Journal of Cancer, 111, 148–154.

[4] Kittler, H., Pehamberger, H., Wolff, K., & Binder, M. (2002). Diagnostic accuracy of dermoscopy. The Lancet Oncology, 3(3), 159–165.

[5] Barata, C., Celebi, M. E., & Marques, J. S. (2018). A survey of feature extraction in dermoscopy image analysis of skin cancer. IEEE Journal of Biomedical and Health Informatics, 23(3), 1096–1109.

[6] Mahbod, A., Schaefer, G., Wang, C., Ecker, R., & Ellinger, I. (2019). Transfer learning using a multi-scale and multi-network ensemble for skin lesion classification. Computer Methods and Programs in Biomedicine, 184, 105138

[7] Codella, N., Rotemberg, V., Tschandl, P., Celebi, M. E., Dusza, S., Gutman, D., ... & Halpern, A. (2019). Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the International Skin Imaging Collaboration (ISIC). arXiv preprint arXiv:1902.03368.

[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778).

[9] Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In International Conference on Learning Representations (ICLR).

[10] M. E. Celebi *et al*., "Dermoscopy image analysis: Overview and future directions," *IEEE Journal of Biomedical and Health Informatics*, vol. 20, no. 5, pp. 1377–1390, 2016.

[11] Y. Yuan, M. Chao, and Y. Lo, "Automatic skin lesion segmentation using deep fully convolutional networks with Jaccard distance," *IEEE Trans. Med. Imaging*, vol. 36, no. 9, pp. 1876–1886, 2017.

[12] L. Yu, H. Chen, Q. Dou, J. Qin, and P. A. Heng, "Automated melanoma recognition in dermoscopy images via very deep residual networks," *IEEE Trans. Med. Imaging*, vol. 36, no. 4, pp. 994–1004, 2017.

[13] M. Combalia and V. Vilaplana, "Monte Carlo dropout uncertainty estimation and loss calibration in skin lesion classification," *Computers in Biology and Medicine*, vol. 144, p. 105367, 2022.

[14] G. J. S. Litjens *et al*., "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, pp. 60–88, 2017.

[15] Y. Kawahara, A. BenTaieb, and G. Hamarneh, "Deep features to classify skin lesions," in *IEEE Int. Symp. Biomedical Imaging* (ISBI), pp. 1397–1400, 2016.