Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.4, August 2025, pp. 1569~1577
e-ISSN: 2548-6861

Implementation of ResNet-50-Based Convolutional Neural Network For Mobile Skin Cancer Classification

Asriani [1]*, Nouval Trezandy Lapatta[2]*, Deny Wiria Nugraha [3]*, Amriana[4]*, Wirdayanti [5]*
* Informatics Engineering, Faculty of Engineering, Tadulako University
asrianiasrianni1213@gmail.com [1], nouval@untad.ac.id [2], deny.wiria.nugraha@untad.ac.id [3]
amriana@untad.ac.id [4] wirda_arbie@untad.ac.id [5]

Article Info

Article history:

Received 2025-06-01
Revised 2025-06-30
Accepted 2025-07-03

Keyword:

Convolutional Neural Network (CNN),
ISIC,
Skin Cancer,
ResNet-50.

ABSTRACT

The skin is one of the most important parts of the human body, serving vital functions such as protecting internal organs from injury, shielding against direct bacterial exposure, regulating body temperature, and more. However, the skin is also susceptible to diseases, one of which is skin cancer. Skin cancer can be extremely dangerous if not treated promptly, as it can lead to death. Therefore, early detection is crucial. This study proposes a technology-based solution by classifying skin cancer using a convolutional neural network (CNN) with a ResNet50 architecture implemented into a mobile application via a REST API using Flask. The HAM10000 dataset, consisting of 10,015 skin lesion images across seven classes, was used for model training. Various testing scenarios were conducted to determine the optimal parameter combination. The best results were achieved with an accuracy of 83.84%, precision and recall of 83%, and an F1-score of 83%, using a training data configuration of 70%, dropout of 0.4, and a batch size of 64. The model implemented in this Android application can perform early detection of skin cancer quickly, practically, and easily accessible to the general public, though healthcare professionals must still supervise it. However, although this model can assist users in making early predictions, the prediction results from this model are only a tool for early detection and do not replace clinical diagnosis by professional medical personnel.2) Figure 8 shows the display for taking pictures through the gallery or camera. Users can choose the image they want to upload from the gallery or the camera to be analysed and predicted by the model.

I. Introduction

As the body's largest organ, the skin is the first line of defence against external factors like physical injuries, ultraviolet (UV) radiation, and dehydration. Additionally, skin plays essential roles in helping to maintain optimal body temperature, detecting changes in the external environment, and supporting the synthesis of vitamin D through sun exposure. However, excessive exposure to UV radiation can damage skin cell DNA and significantly increase the risk of skin cancer. Skin cancer is the most commonly diagnosed cancer globally and is often caused by prolonged sun exposure. It can be classified into three main types: The three main types of skin cancer include Basal Cell Carcinoma (BCC), Squamous Cell Carcinoma (SCC), and Malignant Melanoma (MM). Melanoma is considered the most life-threatening form because it spreads rapidly to other body parts. In contrast, BCC and SCC are generally treatable when identified early. [1]. Skin cancer is considered one of the most prevalent malignant cancers worldwide, including in Indonesia. [2]. According to WHO data from 2020, nearly ten million cancer cases are diagnosed globally each year, and cancer is among the leading causes of death. In Indonesia, skin cancer ranks third after cervical and breast cancer, accounting for approximately 5.9–7.8% of total annual cancer cases. [3]. Skin cancer develops in the epidermis, the outermost layer of the skin, making it visible to the naked eye. [4]. There are many cases of skin cancer found in various countries, making the diagnosis and treatment of this disease crucial.[5].

In the medical world, skin cancer diagnosis requires a biopsy, in which skin tissue is taken and examined in detail to determine whether it is cancerous or not. However, this process takes a long time and must be performed by a dermatologist, and it is also very expensive.[6]Therefore, innovation is needed to enable early detection in diagnosing skin cancer. Rapid diagnosis will increase the chance of curing cancer by 90%, but if treatment is delayed, the chance of recovery is only around 50%.[7]Skin cancer classification is one measure that can be taken to provide an early diagnosis of cancer. This classification helps predict the class label for samples.[8]Through the classification process, early detection can be carried out, and an accurate diagnosis can be made based on the training data. This will help the public identify skin cancer as a common disorder. [6]. This study offers an innovation for classifying skin cancer using the Convolutional Neural Network (CNN) method with the ResNet-50 architecture. The ResNet-50 architecture has been proven effective in various image processing applications, including skin cancer detection.

Previous studies have proven the effectiveness of the Convolutional Neural Network (CNN) algorithm in skin cancer classification. For example, a study using a hybrid preprocessing method divided cancer into two categories (malignant and benign) and achieved an accuracy of 78.19% from 1,530 image data points.[9]. Another study using AlexNet architecture on 4,000 images and dividing the data into four classes achieved a very high accuracy of 99.50%.[1]. Meanwhile, a comparison of the ResNet-50 and VGG-16 architectures shows that ResNet-50 is capable of providing higher accuracy, namely 94% compared to VGG-16, which obtained 91% in two classes (benign and malignant).[3]. Other studies have combined CNN-SVM with the VGG-19 and ResNet-50 architectures, but their accuracy results were lower, at 65.33% and 63.67% using the HAM10000 dataset from Kaggle.[10]. A website-based study testing CNN and ResNet-50 achieved a training accuracy of 96.7% and a validation accuracy of 87.3%.[11]. These studies demonstrate high accuracy in skin cancer classification despite using a smaller amount of data and only distinguishing between two to four types of cancer. This is understandable because the fewer classes there are to classify, the simpler the model's task becomes. The model only needs to recognise the differences between two or a few types of cancer whose differences are sufficiently clear, making the model's learning process easier and resulting in higher accuracy. On the other hand, most of these studies are limited to analysis. They are still confined to desktop sites or websites and have not yet integrated skin cancer classification solutions into mobile applications that are easily accessible.

This study will create the best model using the Convolutional Neural Network (CNN) algorithm with the ResNet-50 architecture. The dataset used in this study is public data sourced from Kaggle, namely the HAM10000 (ISIC) dataset, which contains a total of 10,015 images of skin cancer lesions categorised into seven (7) classes. Although the dataset is large and classifies many categories, this will make the model more effective because it is trained under high difficulty levels, as the model must be able to distinguish between more types of cancer that may have similar shapes, colours, or textures. Additionally, the number of data points per class may be imbalanced, causing the model to favour classes with more data and perform less accurately on classes with fewer data points. This research faces an additional challenge as it is aimed at mobile applications, so the model used must be lightweight and efficient to run on smartphones.

Therefore, despite the greater challenges, this study aims to produce a model that is not only accurate but also practical and ready for immediate use by the community. The final result of this study is the ability to detect seven classes of cancer, with the detection process being easy and efficient using mobile devices. This study is superior to previous studies, which only classified two classes and were limited to desktop sites.

II. LITERATURE REVIEW

A. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is an extension of Multilayer Perceptron (MLP) designed for two-dimensional data processing. Due to its high network depth, CNN is a type of Deep Neural network and is widely applied to image data. CNN is part of Deep Learning, which is widely used for image and video processing. CNNs can automatically extract features from images, reducing reliance on manual feature engineering. [12]CNNs have been applied in various fields, such as object detection, face recognition, medical image classification, and more.[12]The advantage of CNNs is their ability to capture spatial relationships in image data using convolutional operations. [13]. CNN consists of several layers that perform image extraction and classification tasks, namely the Convolutional Layer, Activation Function (ReLu), Pooling Layer, and Fully Connected Layer.

1) Convolutional Layer

This layer is the core layer of CNN and is responsible for extracting features from input images using filters (kernels). Each filter is smaller than the original image and slides across the image to generate a feature map, which represents certain characteristics of the image, such as edges, textures, or specific patterns.[14]

2) Activation Function (ReLu)

After the Convolution layer, Activation functions such as Rectified Linear Unit (ReLu) are applied to introduce non-linearity into the network. This is necessary because many phenomena in the real world are non-linear, and these activation functions help CNNs learn more complex representations. [15]

3) Pooling Layer

The Pooling Layer reduces the dimensionality of feature maps while retaining features. A commonly used type of pooling is Max Pooling, which takes the maximum value from a specific region on feature maps.[15]

4) Fully Connected Layer

After the features are extracted, the results are converted into a one-dimensional vector (flattening) and passed on to a fully connected layer. This layer consists of neurons connected to all neurons in the previous layer and is responsible for classification based on the features learned. Typically, at the final classification stage, the Softmax activation function is used, which will generate probabilities for each possible class.[16]

B. Residual Network -50 (ResNet-50)

Residual Network (ResNet) is a Deep Learning architecture designed to solve the vanishing gradient problem in deep neural networks. [17]. This model has several variants based on network depth, such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. [17]. The ResNet-50 architecture is shown in the following figure.

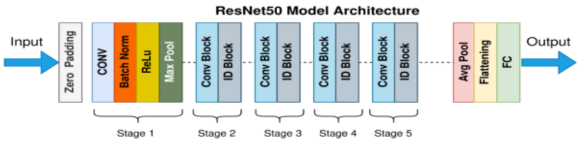


Figure 1. ResNet-50 Architecture

For this study, the researchers chose to use ResNet-50 because this model provides a balance between depth and computational efficiency when compared to other larger ResNet variants. With 50 layers, ResNet-50 is deep enough to extract complex features while maintaining efficiency in devices with limited resources. This model utilises bottleneck residual blocks, which consist of three convolutional layers (1x1, 3x3, 1x1), where the 1x1 convolution serves to reduce the number of parameters, thereby accelerating training. The following table provides a summary of the model structure.

TABLE I
RESNET-50 MODEL STRUCTURE

Layer (type)	Output Shape	Param #
Resnet (Functional)	(None, 7, 7, 2048)	23,587,712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dropout (Dropout)	(None, 2048)	0
dense (Dense)	(None, 1024)	2,098,176
dropout_1 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 7)	7,175

C. Hyperparameter Tuning

Hyperparameter tuning is the process of finding the optimal values for parameters that have not been directly learned by the model during training (in machine learning and deep learning). Hyperparameters greatly influence model performance in terms of accuracy, stability, and training efficiency (Bengio, 2012). Three hyperparameters that are often used in deep learning are learning rate, batch size, and dropout.

1) Learning Rate

The learning rate is a scalar factor that determines how much the weights are updated for each iteration in optimisation algorithms such as Stochastic Gradient Descent (SGD).[18]

2) Batch Size

Batch size is the determination of the number of samples used in one iteration before the weights are updated. The selection of batch size can affect the stability of training and the speed of model convergence.[19]

3) Dropout

Dropout is a regularisation technique that attempts to prevent overfitting by randomly deactivating some neurons during the training process.[20]

D. Flask

Flask is a web framework written in Python and classified as a microframework. Flask functions as a framework for web applications and displays. Using Flask and Python makes it easier for developers to create structured websites and manage web behaviour more easily. [21]. Flask is a bridge for developers to create an API. The Flask API allows developers to create web services that can be accessed by other applications or systems via HTTP, enabling the sharing of data and interaction between System 1 and other systems using standard web protocols. [22].

III. METHOD

Figure 2 shows the flow of the research conducted to build the best training model to be implemented in a mobile application (Android). The first stage is data collection. In this stage, data is taken from the HAM10000 dataset, which is part of the International Imaging Skin Collaboration (ISIC), sourced from Kaggle. Then, we move on to the second stage of data preprocessing, which is preparing the data so that it is ready to be used in training the model. Preprocessing was performed by applying normalisation techniques to the images according to the input format accepted by the ResNet-50 model. Next, the data was divided into three subsets: training (data for training the model), validation (data for monitoring model performance), and testing (data for evaluating model performance). Then, parameter tuning was performed, such as setting the Dropout, Batch Size, and Learning Rate values. These parameters are adjusted to control the model training process and prevent overfitting. These steps are repeated until the best training model is found.

This stage is referred to as the Testing Scenario. Once the best model is obtained, it is implemented into a mobile application via a REST API using Flask, enabling the model to perform instant and efficient early predictions of skin cancer.

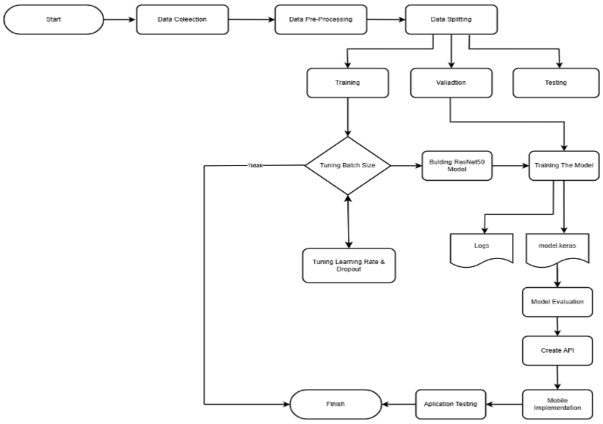
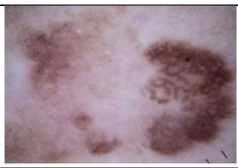
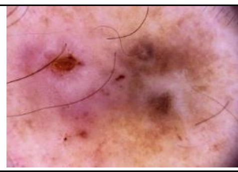


Figure 2. Research Process

A. Dataset

In this study, the dataset used is the HAM10000 Skin Cancer Dataset, which contains images of skin lesions from the International Skin Imaging Collaboration (ISIC). This dataset includes images across seven (7) classes: Actinic Keratoses (AK), Basal Cell Carcinoma (BCC), Benign Keratosis (BK), Dermatofibroma (DF), Melanoma (MEL), Melanocytic Nevi (NV), and Vascular Lesions (VASC), totalling 10,015 image data points. The dataset also includes metadata, diagnostic labels, and lesion properties such as body location and lesion type. Image resolution in the dataset varies from 300x300 to 600x600 pixels. The following image shows the number of lesions distributed across the 7 classes according to their type.

TABLE II
NAME, NUMBER, AND LESION SAMPLE DATASET

No	Type of lesion	Number of lesions	Sample (image of lesion)
1	Actinic Keratoses (AKIEC)	327	
2	Basal Cell Carcinoma (BCC)	514	
3	Benign Keratosis (BKL)	1.099	
4	Dermatofibroma (DF)	115	
5	Melanoma (MEL)	1.113	
7	Vascular Lesions (VASC)	142	

B. Preprocessing

The dataset used in this study contains images with pixel values ranging from 300x300 to 600x600 pixels. The average image has a pixel value of 400x650, which does not match the input of ResNet50, which only accepts inputs of 224 x 224 x 3 pixels. Therefore, data preprocessing is required to adjust the input values accepted by the model. The first step in the preprocessing workflow is to read and organise the dataset containing the metadata. The metadata includes information about the image_id and dx (disease diagnosis label) and the location of the skin lesion. Then, folders are organised based on the label for each disease lesion, which are then arranged in order according to their labels. The images are then moved to the appropriate folders, mapping the images to folders labelled according to the image labels. Next, image normalisation is performed, which is the core stage of preprocessing, as the pixel values of the images are adjusted to match the input required by the ResNet50 model. The data is then divided into three parts: training, validation, and testing, sourced from the folders organised based on skin lesion labels.

Data division is an effort to ensure that the training data is sufficiently diverse and to avoid overfitting on the training data. Next, the model is built using the ResNet-50 architecture, which has undergone training using the ImageNet dataset. The advantage of using a pre-trained model is that it can optimise computational time and help improve model performance. Additionally, ResNet-50 has excellent feature extraction capabilities.

During the model development process, several initial layers of ResNet-50 were frozen to reduce training time and prevent overfitting. These were then replaced to perform specific classification tasks on the skin dataset. The training process also used several techniques, such as GlobalAveragePooling2D, to reduce feature dimensions, as well as dense layers for classification. Meanwhile, optimisation was also carried out using the Adam Optimiser. The training process uses 20 epochs with a callback to stop training early if the model does not show significant improvement. After the training process is complete, the model will be evaluated using several evaluation metrics, such as the confusion matrix and ROC curve, to assess the model's performance and effectiveness. The best model resulting from the ResNet-50 model training will be used for skin disease prediction on Android devices. This implementation will be highly beneficial for efficient and accurate predictions. The model training configuration is summarised in the following table:

TABLE III
MODEL TRAINING CONFIGURATION

Parameter	Nilai
Epoch	20
Optimizer	Adam
Learning rate	0.0001 (1e-4)
Batch size	64
Dropout	0.4
Validation split	0.2 (20%)
Regularisasi L2	0.01

C. Testing Scenario

The classification of skin cancer images using the CNN algorithm with the ResNet50 architecture can be influenced by several factors, such as the amount of data, image quality, data diversity, the percentage of training, validation, and testing data, as well as dropout, batch size, and learning rate in obtaining the best and most efficient model performance. In this study, parameter combinations (dropout, batch size, and data split) were determined using a trial-and-error approach. The researcher tested common values frequently used in CNN model training, such as dropout rates of 0.4 to 0.6 and batch sizes of 16, 32, and 64. Each combination was tested with data splits of 70% and 80%. The results of each combination were compared to select the best configuration based on the highest validation accuracy and loss stability.

TABLE IV
TESTING OF THE 70% DATA DIVISION

Training Data	Dropout	Batchsize	Accuracy (%)	Time
70%	0.4	16	78.00%	48 minutes 34 seconds
		32	83.00%	36 minutes 3 seconds
		64	**83.84%**	**39 minutes 15 seconds**
	0.5	16	81.00%	50 minutes 41 seconds
		32	81.39%	50 minutes 41 seconds
		64	82.37%	50 minutes 41 seconds
	0.6	16	77.81%	50 minutes 41 seconds
		32	80.66%	50 minutes 41 seconds
		64	81.05%	50 minutes 41 seconds

Based on Table IV, the best accuracy obtained from the 70% data testing scenario is 83.84% with a dropout value of 0.4 and a batch size value of 64. This model takes 39 minutes and 15 seconds from initialisation to model evaluation. From the table, it can be seen that as the dropout value increases, there is a possibility of a decrease in model performance. This occurs because, as the dropout value increases, more neurons are deactivated.

TABLE V
TESTING OF THE 80% DATA DIVISION

Training Data	Dropout	Batchsize	Accuracy (%)	Time
80%	0.4	16	78.00%	45 minutes 59 seconds
		32	74.01%	41 minutes 16 seconds
		64	81.85%	38 minutes 23 seconds
	0.5	16	79.21%	49 minutes 38 seconds
		32	80.36%	33 minutes 48 seconds
		64	81.85%	34 minutes 46 seconds
	0.6	16	80.96%	53 minutes 34 seconds
		32	76.69%	33 minutes 44 seconds
		64	**83.35%**	**38 minutes 51 seconds**

From Table V above, it can be seen that the best accuracy produced from the 80% data split is 83.35% with a dropout value of 0.6 and a batch size value of 64. This model requires 38 minutes and 51 seconds from initialisation to model evaluation. From the table, it can be seen that there are several models with high accuracy at certain dropout rates, such as dropout 0.4 with the highest accuracy of 81.85% and a training time of 38 minutes and 23 seconds. Dropout 0.5 also achieved the highest accuracy of 81.85% with a training time of 34 minutes and 46 seconds. From these results, it can be concluded that a batch size of 64 with a dropout rate of 0.6 provides the best training results, although it does require a slight sacrifice in training time.

III. RESULTS AND DISCUSSION

A. Best Model

Based on the tests conducted, the researchers decided to select the best model based on its highest accuracy. From the test scenario table, it can be seen that the highest accuracy was obtained from a 70% data split with a dropout value of 0.4 and a batch size of 64. The following figure shows a graph plot of the Training and validation Accuracy and Loss during the training process of the best model.

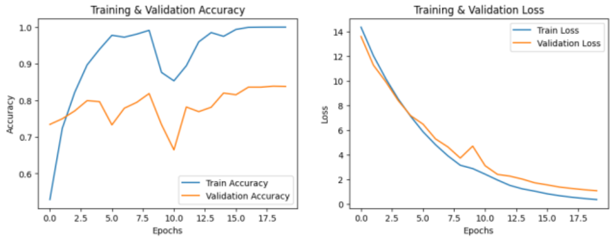


Figure 3. Model Training Log

The figure shows the results of model training over 20 epochs. In the left graph, training accuracy increases sharply to reach 100%, while validation accuracy shows a gradual and stable upward trend in the range of 80%. This indicates that the model is able to learn well from the training data and demonstrates fairly consistent performance on the validation data. On the right graph, both training loss and validation loss show a significant decrease as the number of epochs increases. This decrease indicates that the model progressively improves its ability to minimise prediction errors on both the training and validation datasets. The accuracy achieved by the best model in each class is presented in the table below:

TABLE VI
PER-CLASS MODEL PERFORMANCE EVALUATION

Class	Precision	Recall	F1-Score	Support	AUC/ ROC
akiec	0.48	0.69	0.56	42	0.94
bcc	0.71	0.65	0.68	85	0.93
bkl	0.73	0.67	0.70	174	0.94
df	0.63	0.75	0.69	16	0.98
mel	0.66	0.56	0.61	176	0.91
nv	0.90	0.93	0.91	993	0.94
vasc	0.86	0.95	0.90	20	1.00

As shown in the table, the CNN-ResNet50 model was evaluated to assess its ability to classify seven categories of skin cancer. This model demonstrated excellent performance for Melanocytic Nevi (nv) and Vascular Lesions (vasc), with an F1 score above 0.90 and an AUC value of 1.00 for the vasc class. These results confirm the model's accuracy in identifying these two categories. Meanwhile, other categories such as Benign Keratosis (bkl), Basal Cell Carcinoma (bcc), and Dermatofibroma (df) also produced satisfactory results. Interestingly, even though the df class only had 16 samples, the model was still able to classify them effectively. This is evidenced by an AUC value of 0.98. Meanwhile, the performance for the Melanoma (mel) and Actinic Keratoses (akiec) classes was relatively low compared to other classes, with F1 scores of 0.61 and 0.56, respectively. However, their AUC values remain high (0.91 and 0.94, respectively), indicating that the model can generally distinguish these classes from others. However, threshold adjustments or additional training strategies may be needed to improve its sensitivity in recognising specific cases within these classes.

Based on the overall model evaluation results, the CNN-ResNet50 achieved an accuracy of 83.84%, with precision, recall, and F1-score values of 83% each. These values are weighted averages across all classes, indicating that the model can maintain balanced performance in classifying the seven types of skin cancer tested.

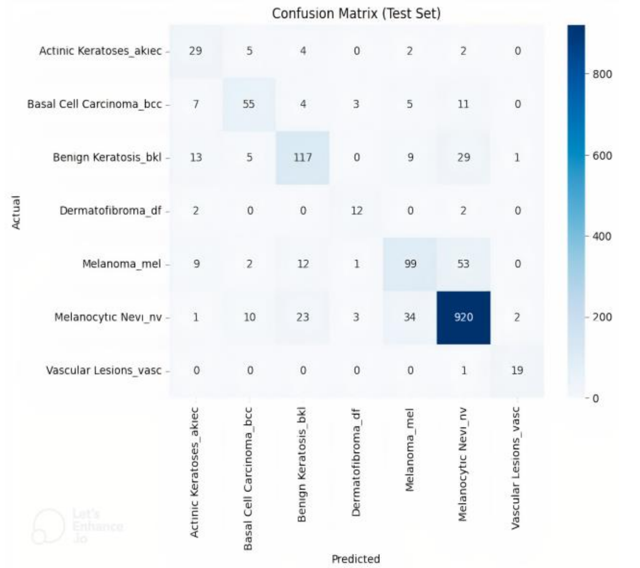


Figure 4. Confusion Matrix Best Model

Figure 4 shows the results of the confusion matrix evaluation, indicating that the classification performs very well for the Melanocytic Nevi (nv) class, with 968 correct predictions. However, the model struggles to distinguish between certain types of skin lesions with visual similarities, such as melanoma, which is often misclassified as the nevus class, and benign keratosis (bkl), which is frequently misclassified as melanoma and nevus. Meanwhile, the VAS class has a high accuracy rate because it only has one prediction error. For classes with a small amount of data, such as df and akiec, the accuracy is low, indicating that the model needs to be improved to handle minority classes effectively.

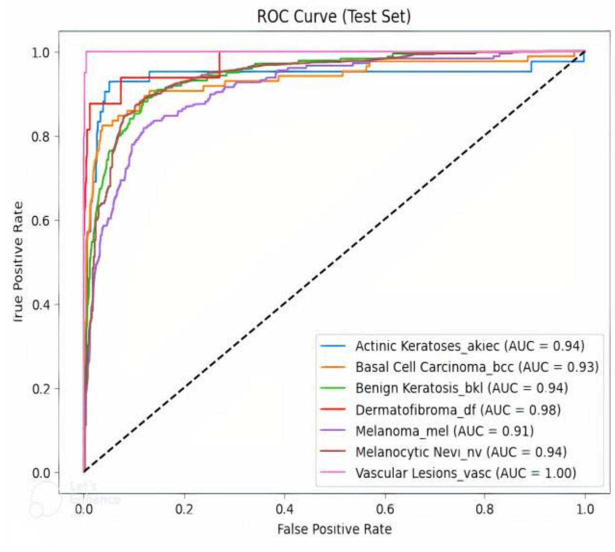


Figure 5. ROC Curve Best Model

Figure 5 shows the ROC curve illustrating the test set results in skin cancer classification using the trained model. The ROC curve illustrates the relationship between the True

Positive Rate (TPR) and False Positive Rate (FPR). TPR represents the model's ability to identify positive cases, and FPR represents the model's error in classifying negative data as positive. The higher the TPR and the lower the FPR, the better the model's performance. In this graph, each skin cancer class, such as Actinic Keratoses_akiec, Basal Cell Carcinoma_bcc, Benign Keratosis_bkl, and others, is represented by a different curve with the AUC (Area Under the Curve) value included. AUC is an important indicator in assessing how well the model distinguishes between positive and negative classes. The AUC value ranges from 0 to 1, with higher values indicating better model performance. From the image, it can be seen that this model provides a very good ROC Curve value (above 0.90) in distinguishing each skin cancer class.

The researchers also conducted studies on other models to assess the efficiency of models from various algorithms, such as MobileNet and GoogleNet Inception V3. The comparison can be seen in the table below.

TABLE VII
MODEL COMPARISON

No.	Model	Result
1.	MobileNet	78.08%
2.	InceptionV3	81.00%
3.	ResNet-50	83.84%

B. *Integrating the model into the Android application*

This stage involves implementing the creation of endpoints from the model created using Flask API. The previously trained model is saved in .h5 format, which is then converted in the API using Flask. The API process can be seen in the following image.

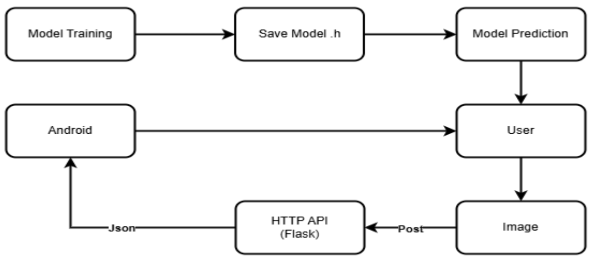


Figure 6. API Process

Figure 6 shows the process that illustrates how the API works in connecting the model and the mobile application so that it can perform image prediction. This process allows users to send images to be predicted by the trained model. This API consists of 2 endpoints that ensure the API runs properly and receives images to be predicted by returning the prediction results. The sent image is first processed to match the input desired by the model, such as image size, format, and normalisation. After that, the model will predict the image class and will return the prediction results along with the confidence or level of trust in the class prediction results.

C. *Application Flow*

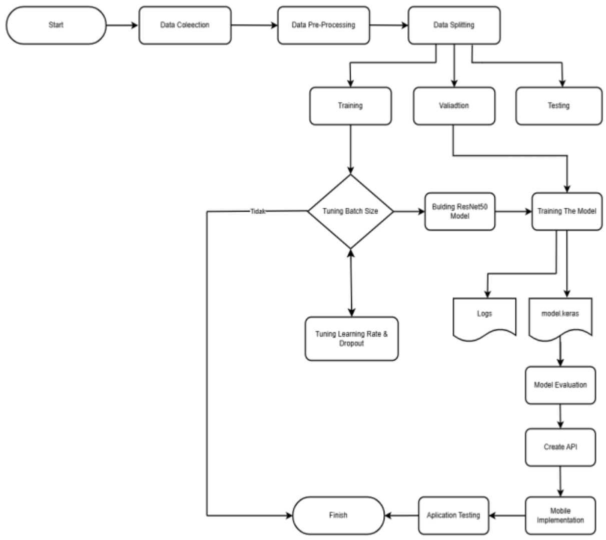
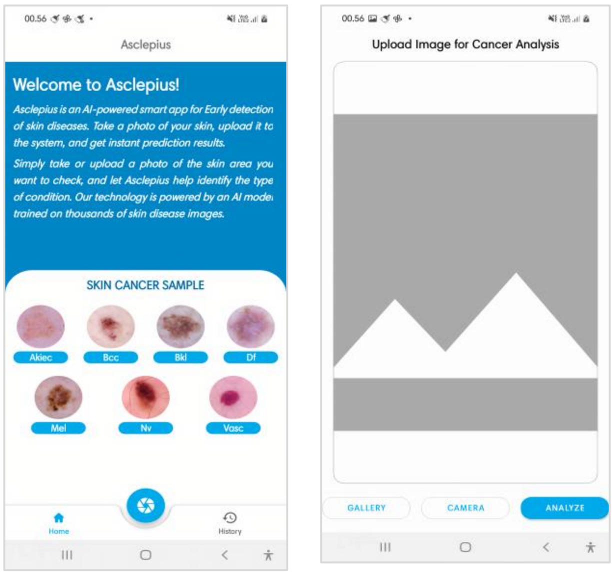


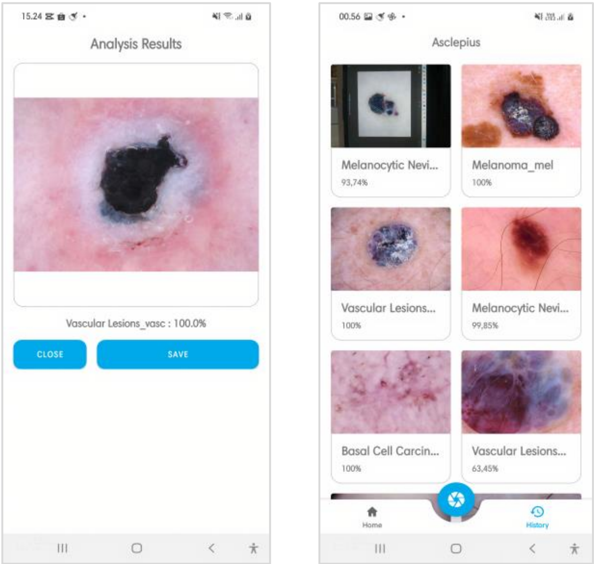
Figure 7. Model Implementation Flow to Mobile Applications

D. *Application display*

The developed mobile application provides a simple interface for loading skin lesion images from a gallery or camera. Once the image is selected, the application sends it to the API server to be processed by the classification model. The classification results, along with the confidence score, are displayed and can be saved in the user's history.



(a) (b)



(c) (d)

Figure 8. illustration of application display (a) home, (b) upload image, (c) result, (d) history.

E. Application Testing

1) Black Box Testing

Testing is a process to check a program to find errors in the program and to test whether the program can run properly according to its function. One of the testing methods that can be done is Black Box Testing. [23]. Blackbox testing is a method that works to check the results based on the input given. Blackbox box testing focuses on the needs of the application. The Blackbox testing process is carried out by trying the program with various inputs available in the application. This testing function determines whether the application that has been created can run according to stakeholder needs. A black box performs checks that are focused on the source code or program module. This testing is only based on external specifications, Such as checking the functionality of the application and observing the basic aspects of the application. [24]

TABEL VIII
PENGUJIAN APLIKASI

No	Function Tested	Expected Result	Status
1.	Function: Displays the Home page containing application information (Home button)	Can display the Home Page	Success
2.	Image Capture Function via Gallery (Gallery Button)	Can Take Pictures Through the Gallery	Success
3.	Image Capture Function via Camera (Camera Button)	Can Take Pictures Through the Camera	Success
4.	The Analyse function on an image is used to make predictions (Analyse button).	Can perform analysis on input images and provide prediction results along with confidence levels for the predictions.	Success
5.	Save Prediction Results Function (Save Button)	Can save prediction results, which are then stored in the history	Success
6.	Close the button if you do not want to save the prediction results	After pressing the Close button, you will return to the Home page.	Success
7.	Prediction History Page (History Button)	Can Display History From Image Prediction Results	Success

2) Prediction Visualisation

Figure 9 shows the prediction results of the CNN-ResNet50 model on 2 two skin lesion images from the HAM10000 dataset. The model classifies the images into Melanoma and Vascular Lesions classes, with a confidence level of 100% for Melanoma and 100% for Vascular Lesions. This visualisation shows that the model is able to recognise the visual characteristics of skin cancer lesions and provide predictions accompanied by probability values, so that it can help users understand the level of system confidence in the classification results provided.

To ensure that the model can run efficiently on mobile devices, performance testing was carried out on real devices using the Samsung Galaxy A71 smartphone. This device has a Qualcomm Snapdragon 730 processor specification, 8 GB of RAM, and the Android 12 operating system.

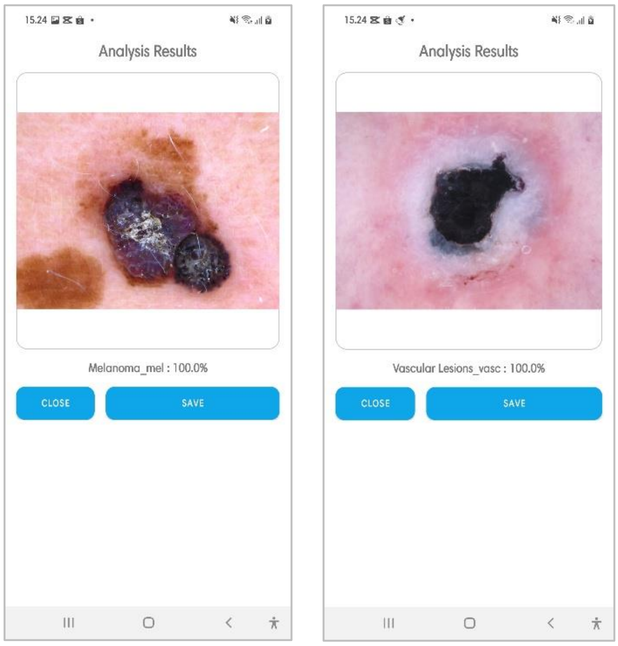


Figure 9. Prediction Visualisation

The test was carried out by accessing the CNN-ResNet50 model that had been trained and saved in .h5 format via the Flask API. Based on the test results, the average inference time to process one image to produce a prediction is around 1.3 seconds under stable network conditions. The size of the

.h5 model file is about 294 MB, and the server API can respond in less than 2 seconds.

IV. CONCLUSION

Based on the results of the research that has been done, it can be concluded that the use of the Convolutional Neural Network (CNN) model with the ResNet-50 architecture is able to provide good performance in classifying skin cancer images into seven classes, namely: Actinic Keratosis (akiec), Basal Cell Carcinoma (bcc), Benign Keratosis (bkl), Dermatofibroma (df), Melanoma (mel), Melanocytic Nevi (nv), and Vascular Lesions (vasc). The model was developed and tested using several scenarios, including variations in data division (70:30 and 80:20) as well as different dropout and batch size settings. From these scenarios, it was found that the best combination was obtained at a data division of 70:30, with a dropout of 0.4, a batch size of 64, and training for 20 epochs. The model managed to achieve the highest accuracy of 83.84%, with precision, recall, and F1-score values of 83% each. These results indicate that the model is able to recognise various types of skin lesions in a balanced manner. In addition, the best model has been successfully implemented into an Android-based mobile application by utilising the API so that it can predict the input image according to the type of lesion given. Although this model can help users to make early predictions, the prediction results of this model are only used as an early detection tool and do not replace clinical diagnosis from professional medical personnel.

REFERENCES

[1] D. A. Nurlitasari, R. Magdalena, and R. Y. N. Fu'adah, "Analisis Performansi Sistem Klasifikasi Kanker Kulit Menggunakan Convolutional Neural Network," *Journal Of Electrical And System Control Engineering*, vol. 5, no. 2, pp. 91–99, Feb. 2022, doi: 10.31289/jesce.v5i2.5691.

[2] Nur Alyyu, "Klasifikasi Kanker Kulit Ganas Dan Jinak Menggunakan Metode Convolutional Neural Network," Dec. 2022. [Online]. Available: https://www.kaggle.com/fanconic/skin-cancer-

[3] Q. Aina Fitroh, "Deep Transfer Learning untuk Meningkatkan Akurasi Klasifikasi pada Citra Dermoskopi Kanker Kulit," 2023.

[4] S. Sa'idah, I. Putu, Y. Nugraha Suparta, and E. Suhartono, "Modifikasi Convolutional Neural Network Arsitektur GoogLeNet dengan Dull Razor Filtering untuk Klasifikasi Kanker Kulit," 2022.

[5] A. Michaelangelo Manurung, I. Santoso, and E. R. Subhiyakto, "The Application of Deep Learning for Skin Disease Classification Using the EfficientNet-B1 Model," 2025. [Online]. Available: http://jurnal.polibatam.ac.id/index.php/JAIC

[6] R. Agustina, R. Magdalena, And N. K. C. Pratiwi, "Klasifikasi Kanker Kulit menggunakan Metode Convolutional Neural Network dengan Arsitektur VGG-16," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 10, no. 2, p. 446, Apr. 2022, doi: 10.26760/elkomika.v10i2.446.

[7] J. T. Sinaga, H. A. Faudyta, and E. R. Subhiyakto, "Klasifikasi Kanker Kulit menggunakan Convolutional Neural Network dengan Optimasi Arsitektur," *Building of Informatics, Technology and Science (BITS)*, vol. 6, no. 3, Dec. 2024, doi: 10.47065/bits.v6i3.6141.

[8] A. W. Kosman, Y. Wahyuningsih, and F. Mahendrasusila, "Pengujian Metode Inception V3 dalam Mengidentifikasi Penyakit Kanker Kulit," *Jurnal Teknologi Informatika dan Komputer*, vol. 10, no. 1, pp. 132–142, Mar. 2024, doi: 10.37012/jtik.v10i1.1940.

[9] F. Dartiko, R. J. Pradana, R. E. Sari, W. Syahputra, W. Kz Oktoeberza, and A. W. Artikel, "Klasifikasi Kanker Kulit Berbasis CNN dengan Metode Hybrid Preprocessing INFO ARTIKEL ABSTRAK", doi: 10.18196/mt.v5i.

[10] M. E. A. R. Ricky Yohannes, "Klasifikasi Jenis Kanker Kulit Menggunakan CNN-SVM," pp. 133–144, Apr. 2022.

[11] D. A. Alfananda and S. R. Nudin, "Implementasi Metode Convolutional Neural Network Dan Resnet Untuk Mendiagnosis Penyakit Kanker Kulit Berbasis Website."

[12] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539ï.

[13] J. Gu *et al.*, "Recent Advances in Convolutional Neural Networks," Dec. 2015, [Online]. Available: http://arxiv.org/abs/1512.07108

[14] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Aug. 01, 2018, *Springer Verlag*. Doi: 10.1007/s13244-018-0639-9.

[15] Y.-L. Boureau, J. Ponce, J. P. Fr, and Y. Lecun, "A Theoretical Analysis of Feature Pooling in Visual Recognition," 2010.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." [Online]. Available: http://code.google.com/p/cuda-convnet/

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition." [Online]. Available: http://image-net.org/challenges/LSVRC/2015/

[18] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimisation," Dec. 2014, [Online]. Available: http://arxiv.org/abs/1412.6980

[19] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On Large-Batch Training for Deep Learning: Generalisation Gap and Sharp Minima," Sep. 2016, [Online]. Available: http://arxiv.org/abs/1609.04836

[20] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," 2014.

[21] R. Y. Azhari, "Technology and Informatics Insight Journal Web Service Framework: Flask dan FastAPI," 2022, [Online]. Available: https://jurnal.universitasputrabangsa.ac.id/index.php/tiij

[22] R. Nandang Pratama and Y. A. Susetyo, "Implementasi Python API dengan Framework Flask sebagai Cloud Run Service Untuk Proses Update di PT. XYZ," 2024.

[23] M. S. Mustaqbal, R. F. Firdaus, and H. Rahmadi, "Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus : Aplikasi Prediksi Kelulusan SNMPTN)," 2015.

[24] B. B. Sasongko, F. Malik, F. Ardiansyah, A. F. Rahmawati, F. Dharma Adhinata, and D. P. Rakhmadani, "Pengujian Blackbox Menggunakan Teknik Equivalence Partitions pada Aplikasi Petgram Mobile," Purwokerto, May 2025.