Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)
Vol. 8, No. 1, March 2025, pp. 249 – 261
p-ISSN: 2614-3372 | e-ISSN: 2614-6150

Automatic Classifier of Road Condition and Early Warning System for Potholes

[1*]Jeremia Manurung, [2]Mansur As, [3]Hamidah Nasution,
[4]Said Iskandar Al Idrus, [5]Kana Saputra S
[1,2,4,5]Department of Computer Science, Universitas Negeri Medan, Indonesia
[3]Department of Mathematics, Universitas Negeri Medan, Indonesia
Email: [1]jeremia.4201250006@mhs.unimed.ac.id, [2]asmansur@unimed.ac.id,
[3]hamidahnst@unimed.ac.id, [4]saidiskandar@unimed.ac.id, [5]kanasaputras@unimed.ac.id

Article Info

Article history:
Received Jul 25th, 2024
Revised Dec 13th, 2024
Accepted Feb 15th, 2024

Keyword:
Android Application
Convolutional Neural Network
Early Warning System
Geographic Information System
Pothole

ABSTRACT

Damaged roads can have a negative impact on road users and can fatally cause accidents. One sign of a damaged road is the presence of holes in the road. This research aims to develop an Android application that can display the location of potholes and provide early warning to driver in Simalungun Regency - North Sumatra. This research implements the Convolutional Neural Network (CNN) algorithm using the transfer learning techniques on the pre-trained MobileNetV3 model for automatic classification of road conditions. The dataset used in the research consisted of 22.538 images which were divided into two classes, namely pothole and normal. This research uses dataset with a ratio of 60:20:20, 70:20:10 and 80:10:10. MobileNetV3 large variant with a dataset ratio of 60:20:20 shows the best value with an F1-Score of 0,9035. The model was further converted to Tensorflow Lite with an F1-Score of 0.8985. This research succeeded in implementing the trained and evaluated model along with early warning of potholes via audiovisual in Android application. Application functionality testing that is carried out using black box testing, showing that the application can run well.

Corresponding Author:
Jeremia Manurung,
Computer Science Student,
Universitas Negeri Medan,
Jl. William Iskandar Ps. V, Kabupaten Deli Serdang, Sumatera Utara 20221, Indonesia.
Email: jeremia.4201250006@mhs.unimed.ac.id

DOI: http://dx.doi.org/10.24014/ijaidm.v8i1.31866

1. INTRODUCTION

Based on statistical data from Statistics Indonesia or Badan Pusat Statistik (BPS) Indonesia, there were 139,258 traffic accidents in Indonesia in 2022 [1]. North Sumatra is one of the provinces with a higher-than-average accident rate in Indonesia, with 6,418 cases accidents occured in 2022 and traffic accident clock 82 minutes which means every 82 minutes an accident happened on North Sumatra [2]. According to the Ministry of Communication and Information of the Republic of Indonesia, the primary cause of traffic accidents is human factors (61%), infrastructure and environmental factors (30%) and vehicle factors (9%) [3].

BPS Simalungun Regency in North Sumatra stated that 54% (971.33 km) of regency roads are severely damaged, 5% (88.83 km) are damaged, 7% (130.77 km) are in moderate condition, and 34% (612.85 km) are in good condition [4]. Based on an interview conducted at Unit Pelayanan Laka Lantas Polres Simalungun on January 12, 2024, at 09:00 AM with IPTU Rudi Junaidi, Head of Law Enforcement Unit, and Bripka M. Tampubolon, S.H., it was estimated that there were approximately 100 accidents caused by damaged roads throughout 2023.

Regarding road repairs to prevent accidents, the Department of Public Works stated that road repairs generally involve three stages: identify, project bidding, and repair [5]. One major issue causing delays in road repairs is the identifying process. According to the Department of Public Works, manually identifying road conditions could takes relatively longer time, approximately 1-2 weeks per kilometer [6]. Due to the lengthy identification process, a new solution that can expedite identifying road conditions with high accuracy is needed.

Various methods have been proposed to create road damage classifiers, including using smartphone accelerometer sensors [7], Light Detection and Ranging (LIDAR) sensors with Arduino Due [8], vibration and ultrasonic sensors with Raspberry PI [9], and using digital camera sensors to capture image and then using methods like Support Vector Machine (SVM) [6], [10] or using deep learning methods based on Convolutional Neural network (CNN) [11], [12], [13].

Compared Naïve Bayes, K-Nearest Neighbor, Support Vector Machine, and Decision Tree algorithms for classifying road damage, indicates that Naïve Bayes with 30% accuracy, K-Nearest Neighbor with 60%, Support Vector Machine with 40%, and Decision Tree with 50%, meaning that these algorithms are less suitable for classifying road damage [10]. In contrast, studies using CNN-based deep learning, such as [11] with SSD-MobileNet achieving 75% recall and precision with 1.5 seconds inference time, [13] use YOLO with 88% accuracy, and [12] create a model with 88% training accuracy and 99% validation accuracy, show that deep learning is suitable for classifying road damage.

Research on creating Geographic Information Systems (GIS) for mapping potholes has been successfully conducted in several cities in Indonesia, such as Malang City [14], South Surabaya [15], and Banjarmasin City [16]. Additionally, systems for reporting road damage and repairs have been developed in the National Road Implementation Center VIII [17] and GIS for roads and bridges in Wonogiri Regency [18]. However, these GIS systems not integrate automatic road condition classifiers yet and are primarily web-based, requiring internet access. These GIS systems also not integrate early system warning yet if the user going to pass the potholes.

From the prior studies, we found several gaps: (1) many existing CNN-based approaches have not been integrated into a full Android-based solution that provides real-time updates and location-based pothole warnings, (2) some advanced CNN architectures (e.g., MobileNetV3) have not yet been widely studied for pothole detection, and (3) early warning features that combine geolocation data with real-time classification remain underexplored. Addressing the aforementioned issues, this research aims to develop an Android application for displaying locations of damaged roads, providing early warnings of road damage, and classifying road conditions based on digital imagery in Simalungun Regency, North Sumatra.

Different from prior studies that use CNN based SSD-MobileNet [11] or YOLO [13], employs MobileNetV3, the latest iteration of the MobileNet architecture, to enhance performance and efficiency. This research starts with implementing a road condition classification model using transfer learning on the pre-trained MobileNetV3 model, an advanced implementation of the CNN algorithm. MobileNetV3, an improvement over MobileNetV2, offers higher accuracy [19], [20], faster inference times, and smaller size [21], making it suitable for low-powered devices like smartphones. Transfer learning will be used to expedite training [20]. The research will continue with developing an Android based GIS application and integrating the previously developed machine learning model, as such a study has not been conducted before.

2. RESEARCH METHOD

The research stages are graphically represented in Figure 1.

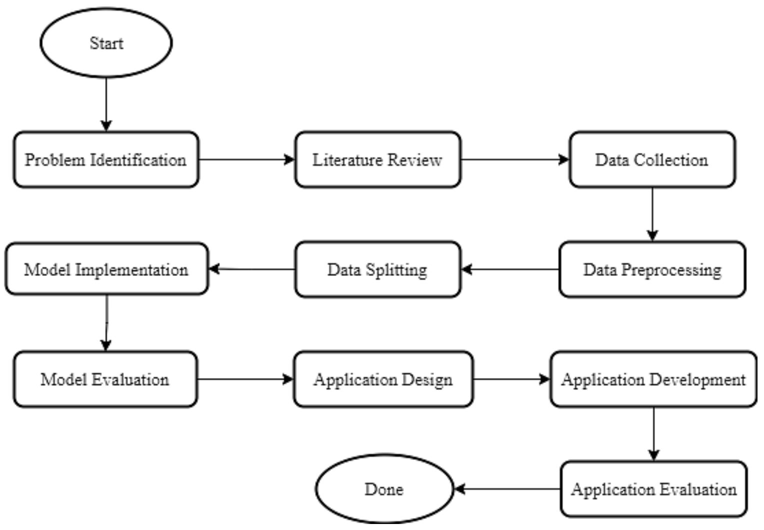


Figure 1. Research Design

2.1. Problem Identification

Problem identification is the initial stage in determining an issue within the research. We gathers information regarding damaged roads in Simalungun Regency and the absence of a geographic information system for these damaged roads.

2.2. Literature Review

At this stage, we conduct a thorough literature review related to the problem and the methods of solution for this research. The review encompasses Geographic Information Systems (GIS), Global Navigation Satellite System (GNSS), Deep Learning, Convolutional Neural Networks (CNN), MobileNetV3, Early Warning Systems, and Android application development. In particular, CNNs are highlighted due to their ability to automatically learn and extract spatial features through convolutional filters—an advantage that makes them particularly effective in tasks requiring image or spatial data processing. Their architecture also enables them to capture hierarchical patterns, which is useful for complex data analysis. Given these strengths, CNNs are well-suited for our research problem, and can be applied to enhance the performance of the Early Warning System, especially within an Android application environment. We searched peer-reviewed journals and major conferences from 2018 to 2024 that used CNN or other machine learning algorithms for pothole (or general road damage) detection. Articles were included if they provided methodological details, reported benchmark results, and addressed real-world feasibility.

2.3. Data Collection

In this stage, primary data is collected using an application that captures images every 2 to 3 seconds. The data collected consists of road images taken using the smartphone's rear camera along with the GPS coordinates of each image. Data collection is conducted on ten road samples located in Simalungun Regency.

2.4. Data Preprocessing

In this stage, we clean the data (delete blurred images and delete still images that were taken while the vehicle wasn't moving) and label the data using the open-source tool "LabelMe." Collected data was then resized to 224x224 pixels, followed by normalization to facilitate more efficient processing by Deep Learning algorithms. Data augmentation is also performed at this stage to create diverse data representations and address class imbalance in the training dataset [22] .

2.5. Data Splitting

The dataset is split with ratios of 70:20:10 (70% training data, 20% validation data, and 10% test data for evaluation), 80:10:10 (80% training data, 10% validation data, and 10% test data for evaluation), and 60:20:20 (60% training data, 20% validation data, and 20% test data for evaluation) to determine the optimal data composition for model training [22], [23], [24].

2.6. Model Implementation

In this stage, we implement the Convolutional Neural Network (CNN) algorithm using transfer learning on the pre-trained MobileNetV3 model both small and large variant. Various hyperparameters are tested during training to identify the model with the highest accuracy [25]. The implemented model classifies two road conditions: potholes and normal.

2.7. Model Evaluation

The model is evaluated using the previously split test dataset. If the model performs poorly at this stage, we will redevelop a new model until optimal performance is achieved. The performance target is 75% for both recall and precision, based on previous research [11]. The model then converted Tensorflow Lite so that the model can run on microcontrollers and smartphone devices.

2.8. Application Design

The research proceeds with application design. This involves creating Unified Modeling Language (UML) diagrams followed by User Interface (UI) design.

2.9. Application Development

The Android application is developed using Android Studio and the Kotlin programming language. We will integrate Google Maps SDK to display maps and locations of damaged road points. Development follows Personal eXtreme Programming (PxP) model. Backend development is also carried out using PHP for login, registration, and classification result sharing functions.

2.10. Application Evaluation

Application evaluation is conducted to ensure that all features are implemented and functioning correctly. The application is tested using the black box testing method.

3. RESULTS AND ANALYSIS

3.1. Data Collection

To create a classifier capable of distinguishing between normal and pothole images, we collected all road images, including those of normal and pothole conditions. The smartphone was mounted on a phone holder horizontally with the rear camera facing the road, as illustrated in Figure 2. The smartphone used for data collection was a Samsung Galaxy S22 Ultra 5G, featuring a main rear camera resolution of 108 Megapixels. Ilustration of Phoneholder position can view figure 2.

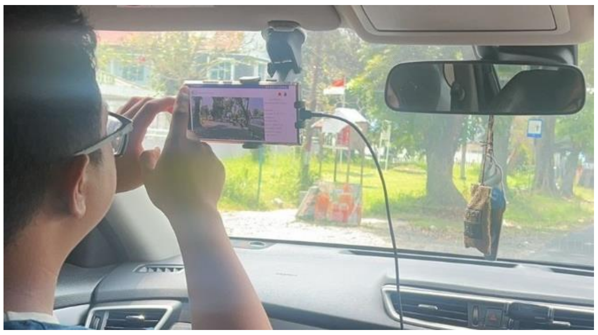
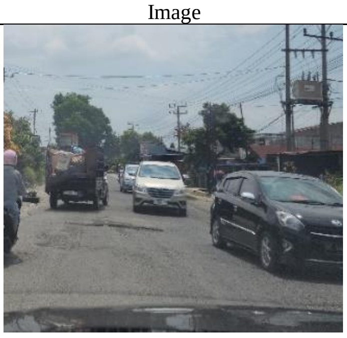
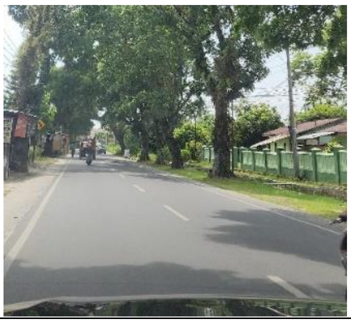


Figure 2. Phoneholder position

We successfully collected 18.000 images while also collected GPS coordinate of the potholes (latitude and longitude). The sample images of potholes and normal road condition are presented in Table 1.

Table 1. Sample image of each class

Class Name	Image
Potholes	
Normal	

3.2. Data Preprocessing

On this stage, we perform data cleaning and data data labeling. the final dataset comprises a total of 15.396 images, with 11.269 images classified as normal and 4.127 images classified as potholes. The distribution across 10 road samples presented in table 2. Although the data presented represents the minimum speed, it is not uncommon for the speed of motorcyclists to reach 99 km/h and 92.52 km/h for cars [26].

Table 2. Distribution of the dataset on 10 Road Samples

No	Street Name	Length (km)	Minimum Speed (km/h)	Total Normal Data	Total Potholes Data	Total Data
1	Jl. Medan - Pematangsiantar (Tebing Tinggi - Pematangsiantar)	22.1	60	1.633	64	1.697
2	Jl. Asahan (Pematangsiantar - Bandar)	43.1	40	1.618	831	2.449
3	Jl. Mayang (Bandar - Raja Maligas)	19.3	20	1.116	653	1.769
4	Jl. Besar Tanah Jawa (Pematangsiantar - Tanah Jawa)	15.6	40	654	194	848
5	Jl. Raya Lintas Utama Sumatera Utara - Tanah Jawa	24.4	20	814	1.717	2.531
6	Jl. Raya Lintas Utama Sumatera Utara - Pematangsiantar	17	60	767	12	779
7	Jl. Pematang Purba - Parapat (Pematangsiantar - Sidamanik)	31	40	1.088	166	1.254
8	Jl. Saribudolok - Tiga Runggu (Sidamanik - Saribudolok)	35.3	40	1.350	41	1.391
9	Jl. Saribudolok (Saribudolok - Merek)	11.4	40	544	19	563
10	Jl. Saribudolok (Pematangsiantar - Raya)	25.4	40	1.685	430	2.115
	Total :			11.269	4.127	15.369

Based on the road status and Peraturan Pemerintah (PP) Number 34 Year 2006.

The total number of standard and pothole images in Table 2 suggests that the collected data is imbalanced, as the number of normal images is nearly three times greater than the number of pothole images. Data mining performed on imbalanced datasets can result in models that perform well in the class with more images but poorly in the class with fewer images [27].

By performing oversampling and image augmentation on the potholes class, we were able to equalize the number of normal and potholes classes, each totaling 11,269 images. The techniques used, including zoom, shift, shear, and horizontal flip, were applied with maximum values of adaptability. These values, which were adapted and modified from previous research [28], [29] demonstrate the flexibility and adaptability of the approach to address data imbalance.

3.3. Data Splitting

At this stage, the dataset was split using three different ratios: 60:20:20 (60% training data, 20% validation data, and 20% test data for evaluation), 70:20:10 (70% training data, 20% validation data, and 10% test data for evaluation), and 80:10:10 (80% training data, 10% validation data, and 10% test data for evaluation) to determine the best data composition for model training [22], [23], [24]. The dataset splitting was performed using the "split-folders" library. Given the large dataset, we opted for a hold-out method rather than cross-validation to reduce computational load, so we can experiment on another model. Detailed data distribution presented in Table 3.

Table 3. Dataset distribution

Ratio	Total Images	
60 : 20 : 20	Train	: 6.761 images each class (13.552)
	Val	: 2.253 images each class (4.506)
	Test	: 2.255 images each class (4.510)
70 : 20 : 10	Train	: 7.888 images each class (15.766)
	Val	: 2.253 images each class (4.506)
	Test	: 1.128 images each class (2.256)
80 : 10 : 10	Train	: 9.015 images each class (18.030)
	Val	: 1.126 images each class (2.252)
	Test	: 1.128 images each class (2.256)

3.4. Model Implementation

The implementation of a CNN for road condition classification was conducted using Transfer Learning techniques. The model utilized was MobileNetV3, with both small and large variants being trained. We implemented MobileNetV3 with an image resolution of 224x224 and used three dataset ratios: 60:20:20, 70:20:10, and 80:10:10. The implemented architecture presented in Figure 3.

Figure 3 shows the structure of MobileNetV3. MobileNetV3 is an enhanced version of MobileNetV2 with squeeze-and-excitation blocks, hard swish activation, and an optimized head and tail design. The architecture begins with a standard convolution with 224x224x3 input, followed by multiple inverted residual blocks (11 blocks on MobileNetV3-Small). Each block uses depthwise convolution layers that significantly reduce computational cost. The final layers include average pooling and convolution 2D, followed by a flatten layer and fully connected layer at the end for classification.

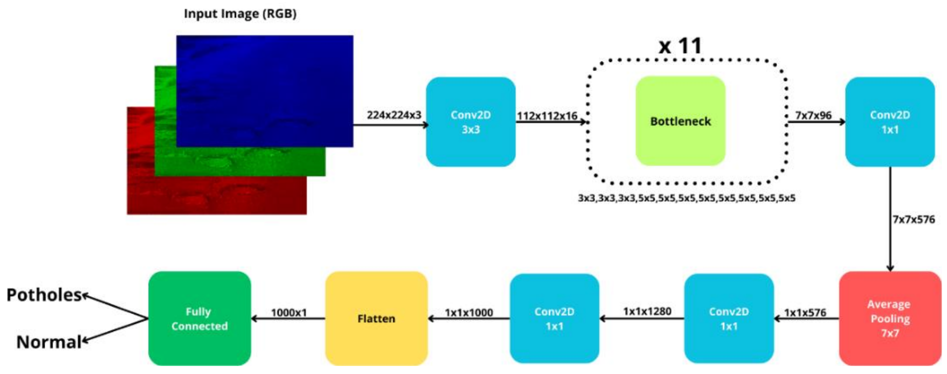


Figure 3. MobileNetV3 architecture

The training result presented in Table 4.

Table 4. Models training result

No	Variant	Dataset Ratio	Total Epoch	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
1	Small	60:20:20	9 / 100	0,8614	0,3294	0,8693	0,3110
2	Large		10 / 100	0,8652	0,3083	0,8853	0,2702
3	Small	70:20:10	6 /100	0,8617	0,3256	0,8868	0,2708
4	Large		8 /100	0,8657	0,3134	0,8937	0,2529
5	Small	80:10:10	7 /100	0,8595	0,3304	0,8859	0,2643
6	Large		11/100	0,8663	0,3121	0,9023	0,2411

To examine the significance between Small and Large variants, we conducted an Independent Sample T-test for the training result. We used the "ttest_ind" function provided by the "scipy" library to calculate the t-statistic and P-value. The Independent Sample T-Test result presented in Table 5. It was concluded that there is a significant difference between the Small and Large models in terms of training accuracy and training loss. However, there is no significant difference in total epochs, validation accuracy, and validation loss.

Table 5. Independent T-Test result for training result

Metrics	t	P-value	Description
Total Epoch	-1.871	0,135	Not Significant
Training Accuracy	-6,415	0,003	Significant
Training Loss	8,127	0,001	Significant
Validation Accuracy	-1,744	0,156	Not Significant
Validation Loss	1,618	0,181	Not Significant

3.5. Model Evaluation

Table 6 shows the performance of the models when tested. It can be observed that the large variant model with a dataset ratio of 60:20:20 outperforms other models in terms of accuracy and F1-Score, while the small variant excels in recall. For the dataset ratio of 70:20:10, the small variant outperforms in precision.

Table 6. Models evaluation result

No	Variant	Dataset Ratio	Accuracy	Precision	Recall	F1- Score
1	Small	60:20:20	0.8778	0.8239	0.9609	0.8872
2	Large		0.9004	0.8759	0.9330	0.9035
3	Small	70:20:10	0.8891	0.8933	0.8838	0.8885
4	Large		0.8980	0.8779	0.9246	0.9006
5	Small	80:10:10	0.8847	0.8715	0.9024	0.8867
6	Large		0.8958	0.8832	0.9122	0.8975

To examine the significance between Small and Large variants, we conducted an Independent Sample T-test for the training result. We used the "ttest_ind" function provided by the "scipy" library to calculate the t-statistic and P-value. The results of the Independent T-test can be seen in Table 7. It can be concluded that there is a significant difference between the Small and Large models in terms of accuracy and F1-Score. However, there is no significant difference in precision and recall.

Table 7. Independent T-Test result for model evaluation

Metrics	t	P-value	Description
Accuracy	-4,004	0,016	Significant
Precision	-0,781	0,478	Not Significant
Recall	-0,315	0,768	Not Significant
F1-Score	-7,205	0,002	Significant

We selected the MobileNetV3 Large variant with a dataset ratio of 60:20:20 as the model to be integrated into the application for road condition classification. This choice was made because the model excels in F1 Score, which is the harmonic mean of recall and precision, providing a balanced view of both metrics. Additionally, the precision and recall values exceed 75%, making this model superior compared to previous studies [11], [30].

The evaluation results in the form of a confusion matrix can be seen in Figure 4. The details are as follows:

1. The True Positive (TP) value in the top left position is 1957, indicating that there are 1957 images of "berlubang" or "potholes" roads correctly predicted as "berlubang" or "potholes" by the model.
2. The False Negative (FN) value in the top right position is 298, indicating that there are 298 images of "berlubang" or "potholes" roads incorrectly predicted as "normal" by the model.
3. The False Positive (FP) value in the bottom left position is 151, indicating that there are 151 images of "normal" roads incorrectly predicted as "berlubang" or "potholes" by the model.
4. The True Negative (TN) value in the bottom right position is 2104, indicating that there are 2104 images of "normal" roads correctly predicted as "normal" by the model.

From this figure, it can be concluded that the resulting model is quite good at detecting images of both "berlubang" or "potholes" and normal roads. However, further research is necessary to reduce the False Negative (FN) value, where "berlubang" or "potholes" road images are misclassified as normal.

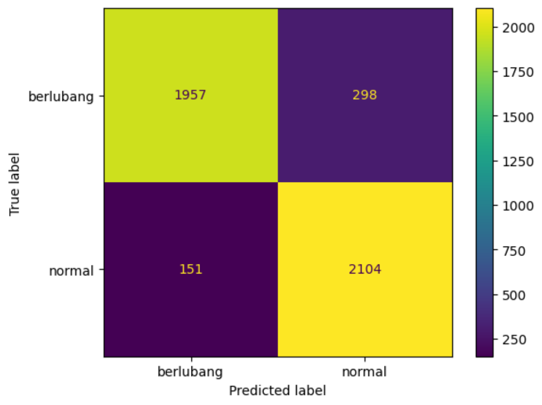


Figure 4. Confusion Matrix

To run the TensorFlow model on a smartphone, microcontroller, or embedded devices, it must first be converted into a TensorFlow Lite model. We converted the TensorFlow ".keras" model into a TensorFlow Lite ".tflite" model using the TensorFlow library. We created two models: one TensorFlow Lite model without optimization and one TensorFlow Lite model with optimization. The comparison between the TensorFlow Lite model without optimization and the TensorFlow Lite model with optimization can be seen in Table 8.

Table 8. Comparison of Tensorflow and Tensorflow Lite models

Metrics	Tensorflow	Tensorflow Lite without optimization	Tensorflow Lite with optimization
Size	21,7 MB	20,9 MB	5,56 MB
Accuracy	0,9004	0,9004	0,8935
Precision	0,8759	0,8759	0,8585
Recall	0,9330	0,9330	0,9423
F1-Score	0,9035	0,9035	0,8985

The performance comparison of the TensorFlow Lite models without optimization and with optimization when running on an Android Smartphone can be seen in Table 9. Init is the time required to initialize the model. First Inference is the time required to run the model for the first time. Warmup (avg) is the average time required during the warmup phase. Inference (avg) is the average time required for running the model. The optimization performed is Quantization, which is a strategy that converts 32-bit float values into 8-bit integers. This reduces the computation and storage requirements.

Table 9. Performance comparison of Tensorflow Lite with and without optimization

Metrics	Tensorflow Lite without optimization (ms)	Tensorflow Lite with optimization (ms)
Init	102,851	102,073
First - Inference	614,854	259,981
Warmup (Avg)	614,854	173,706
Inference (Avg)	460,603	99,215

The comparison of Random Access Memory (RAM) usage when running the models on Android can be seen in

Table **10**. Init represents the number of bytes used during the initialization of the model. Overall indicates the total number of bytes used once the performance testing is completed.

Table 10. RAM usage comparison of Tensorflow Lite with and without optimization

Metrics	Tensorflow Lite without optimization (MB)	Tensorflow Lite with optimization (MB)
Init	67,3867	36,2266
Overall	73,6445	49,9688

3.6. Application Design

Figure 5 illustrates the system design to be implemented. The application will connect to the server using an API to send login data, registration data, and additional data on damaged or roads with potholes (road condition classification history). The server will respond by returning login response data, registration response data, and response data for damaged roads.

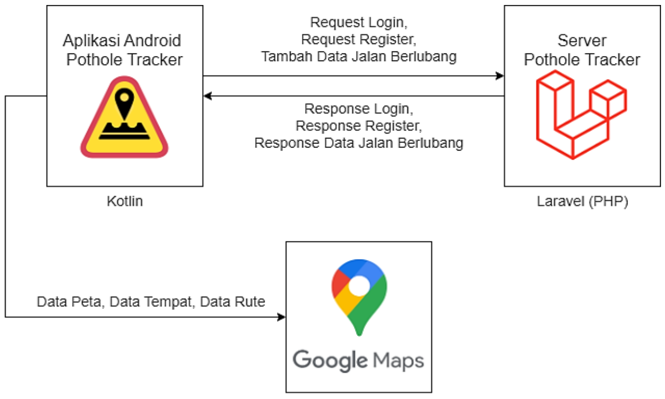
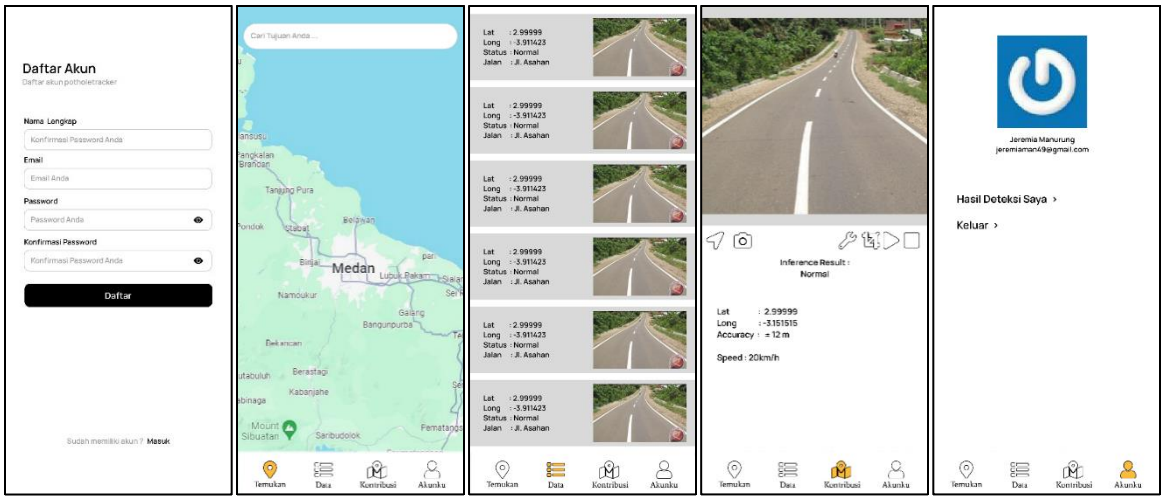

Aplikasi Android Pothole Tracker
Request Login, Request Register, Tambah Data Jalan Berlubang
Server Pothole Tracker
Response Login, Response Register, Response Data Jalan Berlubang
Kotlin
Laravel (PHP)
Data Peta, Data Tempat, Data Rute
Google Maps

Figure 5. System design to be implemented

The interface design was created using Figma to develop the initial design of the application. The designs created with Figma include the main features that will be developed in the application. The User Interface (UI) Design represented in Figure 6.

3.7. Application Development

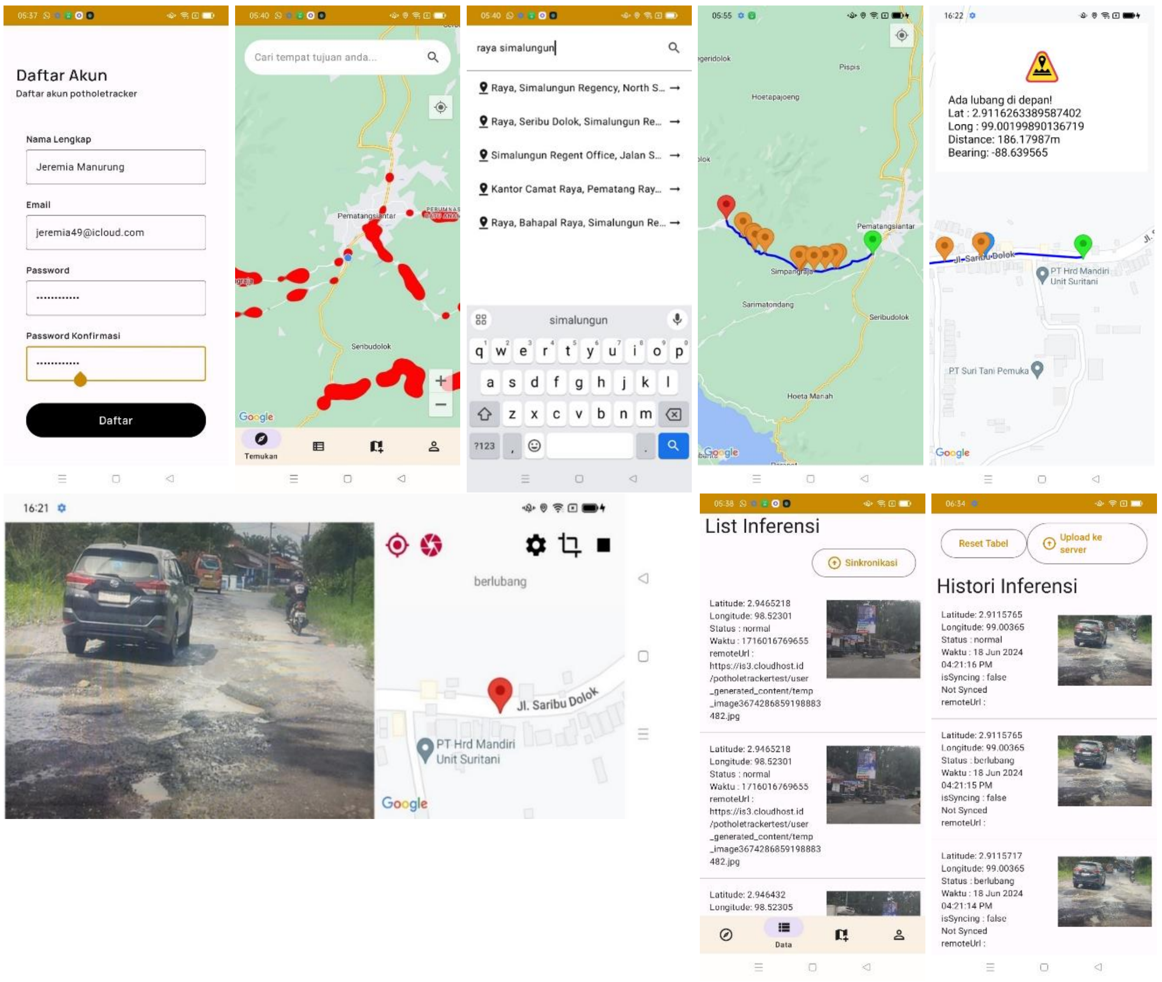
This research developed an Android application named “Pothole Tracker”. The Android application is designed to be run by users on smartphones with the Android operating system. The application uses an API to connect the app with the server. The implementation of the user interface pages was carried out using Jetpack Compose in Android Studio. The screenshot of developed android application represented on Figure 7.



Lat : 2.99999
Long : -3.911423
Status : Normal
Jalan : Jl. Asahan

Lat : 2.99999
Long : -3.151515
Accuracy : ± 12 m
Speed : 20km/h

Figure 6. Application design to be implemented


Daftar Akun
Daftar akun potholetracker
Nama Lengkap
Jeremia Manurung
Email
jeremia49@icloud.com
Password
Password Konfirmasi
Daftar
Cari tempat tujuan anda...
raya simalungun
Raya, Simalungun Regency, North S...
Raya, Seribu Dolok, Simalungun Re...
Simalungun Regent Office, Jalan S...
Kantor Camat Raya, Pematang Ray...
Raya, Bahapal Raya, Simalungun Re...
Pematangsiantar
berlubang
Jl. Saribu Dolok
PT Hrd Mandiri Unit Suritani
List Inferensi
Sinkronikasi
Reset Tabel
Upload ke server
Histori Inferensi
Temukan
Data

Ada lubang di depan!
Lat : 2.9116263389587402
Long : 99.00199890136719
Distance: 186.17987m
Bearing: -88.639565

Latitude: 2.9465218
Longitude: 98.52301
Status : normal
Waktu : 1716016769655
remoteUrl : https://is3.cloudhost.id/potholetrackertest/user_generated_content/temp_image3674286859198883482.jpg

Latitude: 2.9115765
Longitude: 99.00365
Status : normal
Waktu : 18 Jun 2024 04:21:16 PM
isSyncing : false
Not Synced
remoteUrl :

Latitude: 2.9115765
Longitude: 99.00365
Status : berlubang
Waktu : 18 Jun 2024 04:21:15 PM
isSyncing : false
Not Synced
remoteUrl :

Latitude: 2.9115717
Longitude: 99.00365
Status : berlubang
Waktu : 18 Jun 2024 04:21:14 PM
isSyncing : false
Not Synced
remoteUrl :

Figure 7. Screenshot of developed android application

3.8. Application Evaluation

The application testing was conducted using the Blackbox testing method, can view table 11-14.

Table 11. Login Blackbox Testing

No.	Testing Scenario	Test Case	Expected Result	Test Result	Conclusion
1	Entering email and password, then pressing the login button	Email: (correct) Password: (correct)	Redirected to the Home page	As expected	Valid

No.	Testing Scenario	Test Case	Expected Result	Test Result	Conclusion
2	Entering email and password, then pressing the login button	Email: (correct) Password: (incorrect)	Receive a warning that the email/password entered is incorrect	As expected	Valid
3	Entering email and password, then pressing the login button	Email: (incorrect) Password: (incorrect)	Receive a warning that the email/password entered is incorrect	As expected	Valid

Table 12. Register Blackbox Testing

No.	Testing Scenario	Test Case	Expected Result	Test Result	Conclusion
1	Entering name, email, password, and password confirmation	Name: (correct input) Email: (correct input) Password: (correct input) Password Confirmation: (correct input)	Redirected to the login page.	As expected	Valid
2	Entering name, email, password, but with a different password confirmation	Name: (correct input) Email: (correct input) Password: (correct input) Password Confirmation: (incorrect input)	Warning that the passwords do not match.	As expected	Valid
3	Entering name, password, and password confirmation using an email that is already registered	Name: (correct input) Email: (already registered) Password: (correct input) Password Confirmation: (correct input)	Warning that the email is already in use.	As expected	Valid

Table 13. Automatic Classification Blackbox Testing

No.	Testing Scenario	Test Case	Expected Result	Test Result	Conclusion
1	User presses the crop button and sets the image cropping area.	-	The image displayed in the preview matches the cropped area.	As expected	Valid
2	User presses the start button.	-	The application captures the image, performs classification, and saves the image to the database.	As expected	Valid
3	User presses the stop button.	-	The application stops the previous operation.	As expected	Valid
4	User presses the sync button.	-	The application synchronizes the classification history data with the server.	As expected	Valid
5	User presses the Reset Table button.	-	The application deletes the classification history.	As expected	Valid

Table 14. Explore and Early warning system Blackbox Testing

No.	Testing Scenario	Test Case	Expected Result	Test Result	Conclusion
1	Entering a destination and selecting one of the suggested places	Destination: (correct input)	Display the travel route and provide a warning if there are potholes within 100m	As expected	Valid
2	User does not enter a destination but only views the locations of potholes	-	Display the locations of potholes	As expected	Valid

4. CONCLUSION

The results of implementation and evaluation indicate that the deep learning model using transfer learning on the MobileNetV3 model can classify road conditions very well. Out of the six trained models, we selected the MobileNetV3 large variant with a 60:20:20 dataset ratio. The evaluation results for the MobileNetV3 large variant show an accuracy of 0.9004, precision of 0.8759, recall of 0.9330, and F1-Score of 0.9035. After being converted to a Tensorflow Lite model, the accuracy is 0.8935, precision is 0.8585, recall is 0.9423, and F1-Score is 0.8985.

The development of the Android application for classifying road conditions has been successfully implemented and can provide early warnings to drivers proven by all the valid blackbox testing. In the "Home" menu, users can enter a destination, and the application will display the route along with any potholes on that route. If there is a pothole within 100 meters of the user, the application will provide both visual and audio warnings. Additionally, in the "Contribution" menu, users can classify road conditions and assist other users by synchronizing the history of road condition classifications. These features help users anticipate accidents and issues caused by potholes during their journeys.

One strength of this approach is the balance between high accuracy and computational efficiency. The MobileNetV3 Large architecture demonstrates robust performance while maintaining a relatively small model size or footprint, allowing for offline use and minimal hardware requirements. Additionally, the system extends beyond image classification to include user-friendly features such as real-time location mapping and audio-visual alerts, making the solution beneficial for everyday driving scenarios.

Despite these achievements, some weaknesses remain. Notably, false negatives where genuine potholes are misclassified as normal pose a significant safety concern, highlighting the need for further model fine-tuning. Moreover, the hold-out evaluation strategy, chosen to manage computational demands, may not fully generalize to every context as effectively as a cross-validation approach. Finally, the system has been tested mainly in Simalungun Regency, so its applicability to other regions with different road characteristics may require further adaptation.

Looking ahead, several enhancements could strengthen this research. Implementing cross-validation method would provide more comprehensive insights into the model's generalization. Additional sensor data, such as accelerometer readings, could be integrated to complement image-based predictions and minimize false negatives. Moreover, future work may use different model that may further improving the performance.

REFERENCES

[1] BPS, Land Transportation Statistics 2022, 8th ed. Jakarta: BPS Indonesia, 2023. [Online]. Available: https://www.bps.go.id/id/publication/2023/11/27/5a5e4c75e4a25d44b1846446/statistik-transportasi-darat-2022.html

[2] Pusiknas Polri, Jurnal Pusat Informasi Kriminal Nasional Tahun 2022, Edisi 2023. Jakarta: Pusiknas Polri, 2023.

[3] Kementerian Komunikasi dan Informatika Republik Indonesia, "Rata-rata Tiga Orang Meninggal Setiap Jam Akibat Kecelakaan Jalan." Accessed: Jan. 18, 2024. [Online]. Available: https://www.kominfo.go.id/index.php/content/detail/10368/rata-rata-tiga-orang-meninggal-setiap-jam-akibat-kecelakaan-jalan/0/artikel_gpr

[4] D. Manik et al., Kabupaten Simalungun Dalam Angka 2023, 2023rd ed. Simalungun: BPS Kabupaten Simalungun, 2023.

[5] P. Hidayatullah, F. Ferizal, R. H. Ramadhan, B. Qadarsih, and F. Mulyawan, "Pendeteksian Lubang Di Jalan Secara Semi-Otomatis," Sigma-Mu, vol. 4, pp. 41–51, 2012.

[6] Y. Sari, A. R. Baskara, P. B. Prakoso, and M. A. Rahman, "Application of Active Contour Model on Image Processing for Detection of Road Damage," Jurnal Jalan Jembatan, vol. 38, no. 2, pp. 138–147, 2021.

[7] R. Hartono, Y. Wibisono, and R. A. Sukamto, "Damropa (Damage Roads Patrol): Aplikasi Pendeteksi Jalan Rusak Memanfaatkan Accelerometer pada Smartphone," Open Science Framework, no. October, pp. 1–6, 2017, doi: 10.31219/osf.io/yekpr.

[8] L. Drajanta and M. Rivai, "Sistem Pendeteksi Tingkat Kekasaran Permukaan Jalan Menggunakan LIDAR dan Arduino Due," Jurnal Teknik ITS, vol. 8, no. 1, pp. 8–11, 2019, doi: 10.12962/j23373539.v8i1.42463.

[9] H. I. Syah, F. Pradana, and ..., "Pengembangan Sistem Pelaporan Kerusakan Jalan Otomatis Berbasis Sistem Embedded," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 3, no. 3, pp. 2187–2193, 2019, [Online]. Available: http://download.garuda.kemdikbud.go.id/article.php?article=871702&val=10384&title=Pengembangan Sistem Pelaporan Kerusakan Jalan Otomatis Berbasis Sistem Embedded

[10] A. N. Utomo, "Analisa Data Ekstraksi Ciri Citra Momen Histogram dan Perbandingan Model Algoritma Klasifikasi Naive Bayes, Nearest Neighbor, Support Vector Machine, dan Decision Tree Pada Studi Kasus Citra Jalan Aspal Rusak Dan Jalan Aspal Tidak Rusak," Incomtech, vol. 9, no. 2, pp. 8–18, 2020.

[11] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiyama, and H. Omata, "Road Damage Detection Using Deep Neural Networks with Images Captured Through a Smartphone," Computer Aided Civil and Infrastructure Engineering, vol. 33, no. 12, pp. 1127–1141, 2018, doi: 10.1111/mice.12387.

[12] A. Riyandi, T. Widodo, and S. Uyun, "Classification of Damaged Road Images Using the Convolutional Neural Network Method," Telematika, vol. 19, no. 2, p. 147, 2022, doi: 10.31315/telematika.v19i2.6460.

[13] B. Sasmito, B. H. Setiadji, and R. Isnanto, "Deteksi Kerusakan Jalan Menggunakan Pengolahan Citra Deep Learning di Kota Semarang," Teknik, vol. 44, no. 1, pp. 7–14, 2023, doi: 10.14710/teknik.v44i1.51908.

[14] I. R. Zunaidi, T. Afirianto, and K. C. Brata, “Sistem Pemetaan Geografis Jalan Rusak Berbasis Mobile Menggunakan Location Based Service Studi Kasus Kota Malang,” Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 3, no. 10, pp. 10216–10224, 2019.

[15] D. W. S. Pratama and A. W. Utami, “Rancang Bangun Sistem Informasi Geografis Pemetaan Jalan Berlubang Wilayah Surabaya Selatan (Studi Kasus : PT.Binamarga Surabaya),” Jurnal Manajemen Informatika, vol. 6, no. 1, pp. 117–121, 2016.

[16] M. I. Zulkipli, S. Agus, and Firdaus, “Sistem Informasi Geografis Berbasis Web Pemetaan Penanganan Jalan Berlubang Di Kota Banjarmasin,” 2022. [Online]. Available: http://eprints.uniska-bjm.ac.id/9618/

[17] L. P. Sumirat, D. Cahyono, and A. A. Akbar, “Sistem Informasi Geografis Pelaporan Kerusakan Dan Perbaikan Jalan di Balai Besar Pelaksanaan Jalan Nasional VIII Berbasis Web Dan Android,” Jurnal Sistem Informasi dan Bisnis Cerdas, vol. 14, no. 1, pp. 27–36, 2021, doi: 10.33005/sibc.v14i1.2404.

[18] H. B. Alim, “Sistem Informasi Geografis Jalan dan Jembatan di Kabupaten Wonogiri,” UIN SUNAN KALIJAGA, 2016.

[19] J. C. F. da Silva, M. C. Silva, E. J. S. Luz, S. Delabrida, and R. A. R. Oliveira, “Using Mobile Edge AI to Detect and Map Diseases in Citrus Orchards,” Sensors, vol. 23, no. 4, 2023, doi: 10.3390/s23042165.

[20] A. Jakaria and H. F. Pardede, “Comparison of Classification of Birds Using Lightweight Deep Convolutional Neural Networks,” Jurnal Elektronika dan Telekomunikasi, vol. 22, no. 2, p. 87, 2022, doi: 10.55981/jet.503.

[21] S. Pérez Arteaga, A. L. Sandoval Orozco, and L. J. García Villalba, “Analysis of Machine Learning Techniques for Information Classification in Mobile Applications,” Applied Sciences (Switzerland), vol. 13, no. 9, 2023, doi: 10.3390/app13095438.

[22] D. Luthfy, C. Setianingsih, and M. W. Paryasto, “Indonesian Sign Language Classification Using You Only Look Once,” e-Proceeding of Engineering, vol. 10, no. 1, pp. 454–459, 2023.

[23] R. Oktafiani, A. Hermawan, and D. Avianto, “Pengaruh Komposisi Split data Terhadap Performa Klasifikasi Penyakit Kanker Payudara Menggunakan Algoritma Machine Learning,” Jurnal Sains dan Informatika, vol. 9, no. 1, pp. 19–28, 2023, doi: 10.34128/jsi.v9i1.622.

[24] Y. Prabhu and N. Seliya, “A CNN-Based Automated Stuttering Identification System,” in 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), 2022, pp. 1601–1605. doi: 10.1109/ICMLA55696.2022.00247.

[25] S. Lasniari, J. Jasril, S. Sanjaya, F. Yanto, and M. Affandes, “Pengaruh Hyperparameter Convolutional Neural Network Arsitektur ResNet-50 Pada Klasifikasi Citra Daging Sapi dan Daging Babi,” Jurnal Nasional Komputasi dan Teknologi Informasi (JNKTI), vol. 5, no. 3, pp. 474–481, 2022, doi: 10.32672/jnkti.v5i3.4424.

[26] M. Naufal, “Pengaruh Hubungan Geometrik Jalan Raya dengan Tingkat Kecelakaan Ruas Jalan Lintas Sumatera, Aceh, Bireuen, Cot Iju, Paya Meneng, Sp 4 Glee Kapai, Simpang Kameng, Mese,” Universitas Muhammadiyah Sumatera Utara, Medan, 2018. Accessed: Jun. 21, 2024. [Online]. Available: http://repository.umsu.ac.id/bitstream/handle/123456789/8422/SKRIPSI%20MUHAMMAD%20NAUFAL.pdf

[27] M. Bach, A. Werner, and M. Palt, “The proposal of undersampling method for learning from imbalanced datasets,” in Procedia Computer Science, Elsevier B.V., 2019, pp. 125–134. doi: 10.1016/j.procs.2019.09.167.

[28] M. Agil Izzulhaq and Alamsyah, “Penerapan Algoritma Convolutional Neural Network Arsitektur ResNet50V2 Untuk Mengidentifikasi Penyakit Pneumonia,” Indonesian Journal of Mathematics and Natural Sciences, vol. 47, no. 1, pp. 12–22, 2024, [Online]. Available: https://journal.unnes.ac.id/journals/JM/index

[29] T. Bayu Sasongko and A. Amrullah, “Analisis Efek Augmentasi Dataset Dan Fine Tune Pada Algoritma Pre-trained Convolutional Neural Network (CNN),” Jurnal Teknologi Informasi dan Ilmu Komputer, vol. 10, no. 4, pp. 763–768, 2023, doi: 10.25126/jtiik.2023106583.

[30] H. Kusumah, M. R. Nurholik, C. P. Riani, I. Riyan, and N. Rahman, “Deep Learning for Pothole Detection on Indonesian Roadways,” Journal Sensi, vol. 9, no. 2, pp. 175–186, 2023.

BIBLIOGRAPHY OF AUTHORS

Jeremia Manurung is an active undergraduate student from the Computer Science Department, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan.

Mansur AS is an active lecturer at the Computer Science Department, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan. He received his Doctorate Degree in Kyushu University, Japan.

Hamidah Nasution is an active lecturer at the Mathematics Department, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan. She received her Doctorate Degree in Universitas Sumatera Utara.

Said Iskandar Al Idrus is an active lecturer at the Computer Science Department, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan. He received his Master Degree in Institut Teknologi Bandung.

Kana Saputra S is an active lecturer at the Computer Science Department, Faculty of Mathematics and Natural Sciences, Universitas Negeri Medan. He received his Master Degree in Institut Pertanian Bogor.