

Implementation of K-Nearest Neighbour for Automotive Body Surface Defect Detection

Ninuk Wiliani^{*1}, Nabeel Fathurrahman², Herry Wahyono³

1, Universitas Pancasila, Indonesia

2,3, Universitas Krisnadwipayana, Indonesia

*Corresponding author

E-mail address:

ninuk.wiliani@univpancasila.ac.id

Keywords:

Body repair, gray level co-occurrence matrix (GLCM), k-nearest neighbours (KNN)

Abstract

A body repair shop company that handles surface damage to vehicles is very necessary because the exterior appearance of the vehicle plays an important role in creating a good and attractive impression to others. Therefore, body repair companies must be able to adapt and keep up with the times. By combining advanced technology solutions, such as image processing, companies can streamline the identification phase, reduce human error, and optimize the allocation of repair resources. The problem discussed in this study is how to classify the surface of the vehicle body so that the classification model built can increase the accuracy in classifying the surface of the vehicle body. The steps taken in this study are to collect images of the surface of the vehicle's body, and then the image data goes through a preprocessing process using median filtering to remove noise and segmentation techniques to improve image results. After preprocessing, the next step is to extract features based on texture using the Gray Level Co-occurrence Matrix (GLCM) and Statistical methods. Next, the images will be classified using the K-Nearest Neighbors (KNN) method, and the accuracy obtained is 56.80% for $k = 3$. After that, the classification model will be evaluated using the Area Under the Curve (AUC), and the AUC value obtained is 68.75%. With this approach, body repair workshop companies can improve the efficiency and accuracy in classifying vehicle body surfaces by utilizing image processing technology and techniques, allowing for better resource allocation and more reliable repair results.

1. Introduction

Body repair plays a crucial role in maintaining the condition and appearance of a vehicle. Body damage, particularly bumpers, is a common problem that requires precision repair. This repair process generally involves three stages: identification, repair, and finishing. Currently, damage identification often relies on subjective visual inspection [1]. This limitation can lead to misidentification, resulting in failed finishing processes such as peeling paint, blistering, or other defects. These failures not only harm consumers but also damage the company's reputation. To address the issue of subjectivity and improve accuracy, this study proposes a system for automatically identifying defects on car paint surfaces. This system utilizes texture feature extraction using the Gray Level Co-occurrence Matrix (GLCM) and classification using the K-Nearest Neighbor (KNN) algorithm.

GLCM was chosen based on its superior ability to analyze texture features [2]. Defects such as scratches, cracks, and dents have unique and distinct surface texture characteristics. GLCM is capable of quantifying spatial relationships between pixels in an image, thus extracting informative values such as contrast, homogeneity, and correlation that effectively distinguish the texture patterns of each defect type [3][4]. This advantage makes GLCM highly suitable for the task of visual defect identification compared to other methods that may not capture detailed spatial information.

Furthermore, KNN algorithm was chosen as the classification method because of several main advantages relevant to this problem. First, KNN is simple, intuitive, and easy to implement [5][6][7]. Second, KNN is a non-parametric method, meaning it makes no assumptions about the data distribution, making it more flexible for real-world defect data that may not follow a specific distribution. Its main advantage is its effectiveness in classifying objects based on feature proximity [8][9]. Once GLCM successfully extracts the unique features of each defect, KNN can reliably classify new defects into the most appropriate category by comparing them with existing data.

This study aims to distinguish defect characteristics using GLCM feature extraction, applying the KNN algorithm to classify defect types (scratches, cracks, and dents) based on these features, and measuring the accuracy level of the classification model using Confusion Matrix and Area Under Curve (AUC) evaluation [1]. By automating

defect identification objectively, this research is expected to increase the efficiency and quality of the results in the body repair industry.

2. Research Method

The system design conducted in this study consists of 4 stages, namely Business Understanding, Data Understanding, Data Preparation, and Modeling. Each of these stages can be described as follows. These stages can be seen in Figure 1.

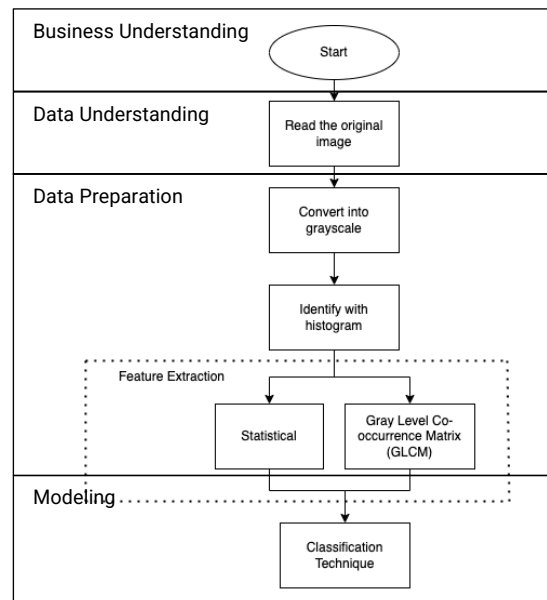


Figure 1. Research method

2.1 Business Understanding

This study focuses on the identification and classification of car body conditions to distinguish between good and defective states. Data collection is performed by capturing images of various car body conditions, which are then compiled into a dataset. This dataset is grouped into two main categories: images of car bodies in good condition and images of car bodies with defects. The 'good condition' category is defined as a smooth body surface without textural damage, while the 'defective condition' category includes visible damage such as scratches and cracks [12].

The classification model is developed through a series of stages: data collection, preprocessing, and feature extraction based on the texture analysis of the car body surface [11]. This texture analysis is key to accurately distinguishing between the two categories. The entire research process, from data processing to model development, is implemented using the Python programming language on the Google Collab platform.

2.2 Data Understanding

The data understanding course focuses on manual data collection, which involves converting 200 data into four categories: good, scratch, dents, and crack. Each category has 50 pixels. After this, the image is converted to grayscale, and a threshold is set to prevent foreground from background.

The process of converting the image to grayscale involves several steps. The first step is to convert the image to grayscale. In the grayscale image, each pixel is represented by three colors: red (R), green (G), and blue (B). To convert it to grayscale, each pixel is transformed into a color with a higher color temperature as shown in Table 1.

The second step is thresholding, a segmentation process to separate the main object (defect) from the background in a grayscale image. To determine the optimal threshold value automatically, this study applies the Otsu Method in the third step. This method was chosen because of its adaptive ability to analyze image histograms to find the most effective threshold value for maximizing contrast. This ensures a more accurate segmentation process, especially in conditions where the color distribution in the image is not uniform [13].

After the image is successfully processed, the next step is Data Augmentation. This process is carried out to enrich and increase the volume of training data without the need for manual collection of new data. The goal is to make the learning model more robust and able to recognize objects or patterns with higher accuracy [14]. In its implementation, the dataset, which initially consisted of several subfolders, each containing 200 images, was augmented through

several steps. This process involves transformations such as image rotation by 45 degrees, as detailed in Table 2. These new augmented images are then added back into the appropriate subfolders, effectively increasing the total amount of training data available for training the model.

Table 1. Real images convert to grayscale




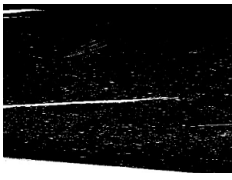

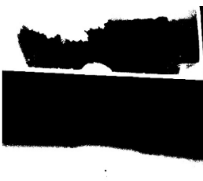

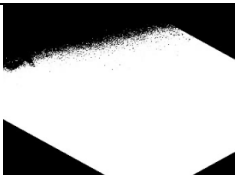
Condition	Real image	Grayscale
good		
Scratch		
Damage		

Table 2. 45° augmentations

Condition	Grayscale	After 45°
good		

2.2 Data Preparation

Data preparation is the process of preparing data for analysis. It involves preprocessing to ensure that the data is suitable for analysis. Median Filtering is a technique used to improve the image quality and reduce noise or interference in the image. This method is used in various image processing applications, such as image restoration, text detection, and noise reduction. Median Filtering works by comparing the intensity of pixels with the median value of pixels in the image [15]. The process includes defining the image's size (window) or kernel; adjusting the size of pixels in the image; adjusting the pixels in the image; and adjusting the median value. The effectiveness of this method in reducing noise, particularly impulsive noise, is also noteworthy.

Segmentation is a process of distinguishing different parts of an image into different areas or objects. Morphology is a technique used to segment the image by identifying the shape or structure of the image. Morphology operates on the morphological element, typically a kernel (also known as a morphology or a structure mask). Kernels are used to perform morphological operations such as erosion, dilatation, formation, and regrowth. The GLCM (Grey Level Co-occurrence Matrix) is used for texture extraction. It analyzes spatial relationships between the intensity of pixels and the color depth (grey level) in the image. Contrast and correlation are two attributes used in this process. Contrast indicates the difference in intensity of pixels, while correlation shows the relationship between the intensity of pixels and the color depth.

Data understanding is crucial for analyzing images and determining the best techniques for enhancing the image quality. By using median filtering, segmentation, and GLCM techniques, researchers can improve the quality of their images and improve the overall quality of their work. The text describes the statistical technique of estimating the mean value of a sample from a dataset using the mean-strategy method. This method uses statistics to analyze the distribution of the sample in the dataset, which can be used to extract data from various objects in the dataset. The three common statistical attributes used in this technique are mean (ratio-ratio), standard deviation, and variance. Mean

is the ratio of the mean value to the sample distribution, which indicates the degree of the sample's gray level. Standard deviation is the degree of variability in the sample distribution, indicating that the sample has a more complex mean value [16]. Variance is the other value of the variability or degree of the sample's gray level.

The process of calculating the mean value from the GLCM and statistical features is divided into two steps: calculating the GLCM features with attributes such as contrast, correlation, energy, and homogeneity; and the statistical features with attributes like mean, standard deviation, and variance. These steps are then combined to create a larger dataset. The GLCM features are applied to the dataset to obtain the corresponding attributes of contrast, correlation, energy, and homogeneity for each sample. The statistical features are applied to the same dataset to obtain the corresponding attributes of mean, standard deviation, and variance [17]. The data is then converted into a CSV file using the functions or libraries provided in the data analysis language.

2.3 Modeling

This study uses a statistical method called KNN to find the most significant values in a dataset. KNN is a function used to calculate the probability of each value from a large dataset. It also performs a comparison between data testing and training. The most significant results are compared to the data that is currently being analyzed. The K-Nearest Neighbor is the output of GLCM, Statistical, and a combination of GLCM and Statistical methods. The K-Nearest Neighbor has the highest probability of each value.

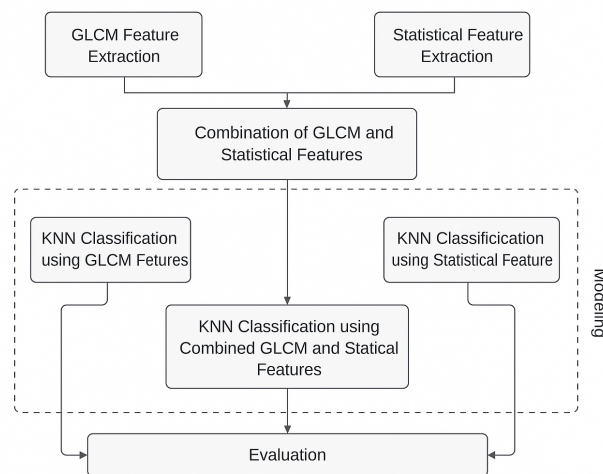


Figure 2. Flowchart modeling

Figure 2 show the research modeling aims to classify the condition of object surfaces based on texture features extracted from digital images. The overall workflow consists of four main stages: feature extraction, feature combination, modeling (classification) and evaluation. Each stage is explained in the next section.

2.3.1 Feature Extraction

a. GLCM (Gray Level Co-occurrence Matrix) Feature Extraction

GLCM is used to capture the spatial relationship between pixel pairs in a grayscale image. From the GLCM, several texture features are derived, including contrast, correlation, energy, and homogeneity. The extraction is performed by setting specific pixel distances (d) and orientations (0° , 45° , 90° , and 135°). Table 3 shows the results of GLCM features.

Index	Contrast	Correlation	Energy	Homogeneity	Label
1	256,1270	0.9919	0.6393	0.9416	Scratch
2	281,3687	0.9911	0.6356	0.9386	Scratch
3	303,6730	0.9905	0.6282	0.9352	Scratch
4	351,6157	0.9889	0.6163	0.9218	Scratch
5	252,7289	0.9921	0.644	0.9473	Scratch

b. Statistical Feature Extraction

Statistical features are calculated based on the intensity distribution of pixels within the image. The indicators include mean, variance, standard deviation, skewness, kurtosis, and entropy, which describe the global characteristics of the image. Table 4 shows the results of statistical features.

Table 4. Result of feature of statistical

Index	Mean	Std	Var	Label
1	114,485	126,3521	15964,859	Scratch
2	115,161	126,416	15981,0702	Scratch
3	122,181	126,877	16097,9668	Scratch
4	141,142	126,122	15906,7922	Scratch
5	121,090	126,924	16109,8948	Scratch

2.3.2 Feature Combination

After the extraction process, features obtained from both GLCM, and statistical methods are combined into a single feature set. This step aims to leverage the strengths of each approach, providing a more comprehensive representation of texture information. Table 5 shows a combination of GLCM and statistical.

Table 5. Combination of GLCM and statistical

Index	Contrast	Correlation	Energy	Homogeneity	Mean	Std	Var	Label
1	256,1270	0.9919	0.6393	0.9416	114,485	126,3521	15964,859	Scratch
2	281,3687	0.9911	0.6356	0.9386	115,161	126,416	15981,0702	Scratch
3	303,6730	0.9905	0.6282	0.9352	122,181	126,877	16097,9668	Scratch
4	351,6157	0.9889	0.6163	0.9218	141,142	126,122	15906,7922	Scratch
5	252,7289	0.9921	0.644	0.9473	121,090	126,924	16109,8948	Scratch

2.3.3 Modeling (Classification using KNN)

To comprehensively evaluate classification performance, this study implements KNN algorithm through three distinct experimental schemes. Each scheme is designed to systematically assess the effectiveness of a specific feature set. Initially, the KNN model is trained and tested using features derived exclusively from GLCM. This approach serves to isolate and analyze the predictive power of textural information alone. Subsequently, a parallel model is developed utilizing only first-order statistical features, such as mean, standard deviation, and skewness, to benchmark their independent performance. Finally, a third scheme investigates the potential for synergistic improvement by employing a hybrid feature vector that integrates both GLCM and statistical features. This final test aims to determine if the fusion of these two feature sets yields a superior classification accuracy compared to using either set in isolation.

3. Results and Discussion

The experiments were conducted using a dataset of [insert dataset size, e.g., 4000 images of solar panel surfaces with four classes: crack, scratch, stain, and normal. The classification process utilized KNN algorithm under three scenarios: GLCM features only, Statistical features only, and Combined features. The results for each scenario are discussed below.

3.1 Classification Using GLCM Features

In this scenario, the KNN model was trained using texture features extracted from the GLCM method. The features included contrast, correlation, energy, and homogeneity, calculated at multiple orientations and distances. The use of odd K values in the KNN algorithm is a common practice that aims to avoid ambiguity or ties during the class determination process.

Table 6. Accuracy with K value

K result	Accuracy (%)
K = 1	47.50 %
K = 3	51.25 %
K = 5	48.75 %
K = 7	50.62 %
K = 9	52.50 %

Table 6 show the model achieved an accuracy of 52.50 %, The relatively high performance indicates that GLCM features effectively capture textural differences between defect types. However, the accuracy was slightly lower for classes with subtle texture variations, such as scratches and stains, likely because GLCM focuses on pixel co-occurrence patterns but does not fully capture global intensity distribution.

3.2 Classification Using Statistical Features

In this scenario, the model utilized mean, variance, standard deviation, skewness, kurtosis, and entropy as the primary features. Table 7 shows the results. The KNN classifier achieved an accuracy of 40.62 %, which is lower

compared to the GLCM-based model. Statistical features provide a global representation of the image but lack spatial information about texture. This limitation affects the model's ability to differentiate classes with similar intensity distributions but different structural patterns.

Table 7. Accuracy with K value

Result of K	Accuracy
K = 1	38.75 %
K = 3	35.00 %
K = 5	38.75 %
K = 7	39.38 %
K = 9	40.62 %

3.3 Classification Using Combined Features (GLCM + Statistical)

In the final scenario, both GLCM and statistical features were combined into a single feature set. Table 8 shows the results. The combined model achieved the highest accuracy of 56.88 %, outperforming the other two scenarios. Improvements were observed in both precision and recall across all classes. The integration of GLCM and statistical features enhances the discriminatory power of the classifier. While GLCM captures spatial texture relationships, statistical features provide complementary global intensity information. This synergy allows for more robust classification, especially for defects with mixed characteristics. The results clearly demonstrate that the combined feature set significantly improves classification performance. This finding suggests that incorporating diverse feature types is essential when dealing with complex texture patterns. However, the increase in dimensionality may lead to higher computational costs, which should be considered for real-time applications.

Table 8. Accuracy with K value

Result of K	Accuracy
K = 1	53.12 %
K = 3	56.88 %
K = 5	50.62 %
K = 7	55.62 %
K = 9	53.75 %

3.4 Evaluation

A confusion matrix is an evaluation method used to assess the accuracy of a classification model. It provides a summary of the predictions made by the model compared to the actual ground truth values. By analyzing the confusion matrix, various performance metrics can be derived, such as accuracy, precision, recall, and F1-score. The classification results from all three scenarios are evaluated using performance metrics such as accuracy, precision, recall, and F1-score. This evaluation aims to compare the performance of each scenario and identify the best-performing approach.

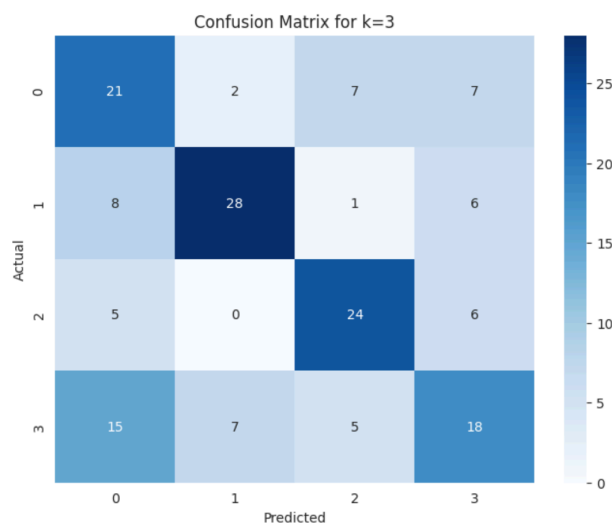


Figure 3. Confusion matrix for K = 3

These metrics offer insights into the model's performance for each class and help evaluate its overall effectiveness in classification tasks. In this study, a total of 800 data samples were used, with 20% of the data allocated for testing, resulting in 160 test samples.

Table 9. Actual versus prediction

Class	Actual	Prediction			
		Crack	Dents	Good	Scratch
Crack	40	21	2	7	7
Dents	40	8	28	1	6
Good	40	5	0	24	6
Scratch	40	15	7	5	18
Total	160	49	37	37	37

Table 9 showed forty real-life 'Crack' examples exist. Out of these, the model accurately identified 21 instances as 'Crack', but misidentified two as 'Smooth'. Seven instances are labeled 'Scratch', and seven are labeled 'Dents'. Forty real 'Dent' examples are included. Out of that total, the model accurately identified 8 instances as 'Dents', however it misidentified 28 as 'Smooth', 1 as 'Scratches', and 6 as 'Cracks'. There are forty real-life instances of 'Smooth'. Of those, 24 cases were accurately predicted by the model to be 'Smooth', while 5 examples were wrongly projected to be 'Cracks', and 6 examples 'Scratches'. There are forty real-life instances of 'Scratches'. Out of that total, the model accurately identified 18 instances as 'Scratches', but misidentified 15 as 'Cracks', 7 as 'Dents', and 5 as 'Smooth'.

Table 10. Actual versus prediction

Class	True positive (TP)	False positive (FP)	False negative (FN)	True negative (TN)
Crack	21	28	16	95
Dents	28	9	15	123
Good	24	13	11	123
Scratch	18	19	27	123

Table 10 showed forty real-life 'Crack' examples exist. Out of these, the model accurately identified 21 instances as 'Crack' but misidentified two as 'Smooth'. Seven instances are labeled 'Scratch', and seven are labeled 'Dents'. Forty real 'Dent' examples are included. Out of that total, the model accurately identified 8 instances as 'Dents'. However, it misidentified 28 as 'Smooth', 1 as 'Scratches', and 6 as 'Cracks'. There are forty real-life instances of 'Smooth'. Of those, 24 cases were accurately predicted by the model to be 'Smooth', while 5 examples were wrongly projected to be 'Cracks', and 6 examples 'Scratches'. There are forty real-life instances of 'Scratches'. Out of that total, the model accurately identified 18 instances as 'Scratches', but misidentified 15 as 'Cracks', 7 as 'Dents', and 5 as 'Smooth'.

$$Recall = \frac{TP}{TP + FN} \quad (1)$$

$$Total = \frac{Crack + Dents + Good + Scratch}{Class} \quad (2)$$

Table 11. Count of recall

Recall	
$Recall (Crack) = \frac{21}{21 + 16} = 0,5676$	$Recall (Dents) = \frac{28}{28 + 15} = 0,6512$
$Recall (Good) = \frac{24}{24 + 11} = 0,6857$	$Recall (Scratch) = \frac{18}{18 + 27} = 0,4000$
$Total = \frac{0,5676 + 0,6512 + 0,6857 + 0,4000}{4} = 0,5761$	

The recall (sensitivity) calculation for each class in the classification model is displayed in the table. The percentage of positive occurrences (True Positives) that the model accurately detects is known as recall. Recall is calculated using the formula True Positive (TP) / (True Positive (TP) + False Negative (FN)).

The 'Crack' class recall is approximately 0.5676. This indicates that the model correctly classified roughly 56.76% of all cases that are 'Crack' as such. The 'Dents' class recall is approximately 0.6512. This shows that, out of all the cases that belong to that class, the model correctly identified roughly 65.12% of them as 'Dents'. The 'Smooth' class recall is around 0.6857. This indicates that out of all the examples, the model was able to identify roughly 68.57% of

them as being genuinely 'Smooth'. The 'Scratch' class recall is approximately 0.4000. This suggests that only roughly 40.00% of occurrences were accurately recognized as 'Scratch' by the model. The average recall, or total recall, is roughly 0.5761. This figure is the mean of all the classes' recall values. It shows that the model effectively detects approximately 57.61% of positive cases in all classes on average. To sum up, the model displays varying recall rates for every class. Recall is lowest in the 'Scratch' class (about 40.00%) and highest in the 'Good' class (approximately 68.57%). According to the Total Recall (Average Recall), the model's overall performance is approximately 57.61%. This indicates that the model may be made more adept at spotting positive examples, particularly for the 'Gores' class, where its recall is quite poor.

$$Specificity = \frac{TN}{TN + FP} \quad (3)$$

$$Total = \frac{Crack + Dents + Good + Scratch}{Class} \quad (4)$$

Table 12. Count of specificity

Specificity	
$Specificity (Crack) = \frac{95}{95 + 21} = 0,8190$	$Specificity (Dents) = \frac{123}{123 + 28} = 0,8146$
$Specificity (Good) = \frac{123}{123 + 21} = 0,8542$	$Specificity (Scratch) = \frac{123}{123 + 18} = 0,8723$
$Total = \frac{0,8190 + 0,8146 + 0,8542 + 0,8723}{4} = 0,8400$	

Table 12 determine specificity for each of the following classes ('Crack', 'Dents', 'Good', 'Scratch') using the following table: Specificity is equal to TN divided by FP. For every grade, TN stands for True Negative and FP for False Positive. Specialization reduces the proportion of actual negative samples that are clearly classified as negative by the model. Regarding 'Crack', the specificity is approximately 0.8190. This means that around 81.90% of the actual 'Crack' samples were successfully classified as such by the model. The specificity for the 'Dents' grade is around 0.8146. This indicates that around 81.46% of the actual 'Dents' samples were successfully identified by the model as 'Dents'. For 'Good', the specificity is approximately 0.8542. This means that around 85.42% of the actual 'Good' samples were successfully classified as such by the model. The specialization for the 'Scratch' grade is around 0.8723. This indicates that around 87.23% of the actual 'Scratch' samples were successfully identified by the model as 'Scratch'. Overall specificity is approximately 0,8400, indicating that the model has an accuracy rate of approximately 84.0% in classifying negative samples for all grades ('Crack', 'Dents', 'Good', 'Scratch'). One of the most important specifications is that the model can correctly identify negative samples, which is a crucial evaluation metric for the classification model.

$$AUC = \frac{0,5761 + 0,8400}{2} = 0,6875$$

Once the Recall and Specificity values have been correctly obtained, they may be entered into the AUC calculation to determine the Area Under the Curve. This is the calculation of the AUC value. The classification task's average performance across all classes is represented by the AUC value. Because it shows a stronger capacity to differentiate between positive and negative examples, a higher AUC value is indicative of a better model. The model's moderate discriminative ability is indicated by the study's computed AUC of 0.6875. This model performs admirably in differentiating between those groups, while not yet achieving the highest level of accuracy.

4. Conclusion

The outcomes of feature extraction using GLCM and statistical feature extraction based on the texture of each image in the dataset are shown in Table 8 on the combined GLCM and statistical feature extraction data, the K-Nearest Neighbour approach successfully diagnosed the types of faults on the automobile body surface with the greatest accuracy value of k, which is 56.88%. The Area Under Curve (AUC) value of the classification model constructed using this method was 68.75%. The model in this investigation has a modest capacity to differentiate, as indicated by its AUC value of 68.75%.

References

- [1] M. H. Prova and Y. Liu, "Understanding cross-functional collaboration in quality work: A case study of hood production at Volvo Cars Body Components," to be published, 2025.
- [2] S. Bakheet and A. Al-Hamadi, "Automatic detection of COVID-19 using pruned GLCM-based texture features and LDCRF classification," *Comput. Biol. Med.*, vol. 137, p. 104781, Oct. 2021.

- [3] S. Barburiceanu, R. Terebes, and S. Meza, "3D texture feature extraction and classification using GLCM and LBP-based descriptors," *Appl. Sci.*, vol. 11, no. 5, p. 2332, 2021.
- [4] M. K. Ghalati et al., "Texture analysis and its applications in biomedical imaging: A survey," *IEEE Rev. Biomed. Eng.*, vol. 15, pp. 222–246, 2021.
- [5] S. S. Priscila, S. S. Rajest, R. Regin, and T. Shynu, "Classification of satellite photographs utilizing the K-nearest neighbor algorithm," *Cent. Asian J. Math. Theory Comput. Sci.*, vol. 4, no. 6, pp. 53–71, 2023.
- [6] P. Cunningham and S. J. Delany, "K-nearest neighbour classifiers—A tutorial," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–25, 2021.
- [7] L. Le, Y. Xie, and V. V. Raghavan, "KNN loss and deep KNN," *Fundam. Inform.*, vol. 182, no. 2, pp. 95–110, 2021.
- [8] M. Auleria, A. I. Arrahmah, and D. E. Saputra, "A review on k-nearest neighbour based classification for object recognition," in *Proc. 2021 Int. Conf. Data Sci. Appl. (ICoDSA)*, Bandung, Indonesia, 2021, pp. 1–6.
- [9] H. Singh, V. Sharma, and D. Singh, "Comparative analysis of proficiencies of various textures and geometric features in breast mass classification using k-nearest neighbor," *Vis. Comput. Ind. Biomed. Art*, vol. 5, no. 1, p. 3, 2022.
- [10] S. Walters, "Area under the curve (AUC)," in *Encyclopedia of Quality of Life and Well-Being Research*. Cham, Switzerland: Springer, 2024, pp. 252–254.
- [11] S. C. Leong, Y. M. Tang, C. H. Lai, and C. K. M. Lee, "Facial expression and body gesture emotion recognition: A systematic review on the use of visual data in affective computing," *Comput. Sci. Rev.*, vol. 48, p. 100545, 2023.
- [12] M. Merino et al., "Body perceptions and psychological well-being: A review of the impact of social media and physical measurements on self-esteem and mental health with a focus on body image satisfaction and its relationship with cultural and gender factors," *Healthcare*, vol. 12, no. 12, p. 1396, 2024.
- [13] Y. Liu, T. Zhang, M. Liang, and E. Wang, "Segmentation of lung nodules in CT images using weighted average-based threshold and maximized variance," *AIP Adv.*, vol. 14, no. 9, p. 095031, 2024.
- [14] T. B. Sasongko, H. Haryoko, and A. Amrullah, "Analisis efek augmentasi dataset dan fine tune pada algoritma pre-trained convolutional neural network (CNN)," *J. Teknol. Inf. Ilmu Komput. (JTIK)*, vol. 10, no. 4, pp. 763–768, 2024.
- [15] N. Wiliani, T. Khawa, and S. Ramli, "Peningkatan kontras pada preprocessing gambar permukaan solar panel dengan histogram," *Innovation and Technology*, vol. 2, no. 1, 2025.
- [16] N. Wiliani, T. Rahman, S. Ramli, and A. Sani, "Statistical characteristics for identification defect of solar panel with Naive Bayes," in *Proc. 2nd Int. Conf. Electr., Telecommun. Comput. Eng. (ELTICOM)*, Medan, Indonesia, 2019, pp. 2–8, doi: 10.4108/eai.27-4-2019.2286885.
- [17] O. Rainio, J. Teuho, and R. Klén, "Evaluation metrics and statistical tests for machine learning," *Sci. Rep.*, vol. 14, no. 1, p. 6086, 2024.