Translation of article 10.22146/jnteti.v13i1.8730

The Best Texture Image for Gaussian Naïve Bayes With Nearest Neighbor Interpolation

Irwan Budi Santoso[1], Shoffin Nahwa Utama[1], Supriyono[1]

[1] Informatics Engineering Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim, Malang, Indonesia

[Submitted: 17 July 2023, Revised: 22 September 2023, Accepted: 13 December 2023]
Corresponding Author: Irwan Budi Santoso (email: irwan@ti.uin-malang.ac.id)

ABSTRACT — One of the factors affecting the performance of the Gaussian naïve Bayes classifier (GNBC) in texture image classification is the image size (dimensions). Image size is one of the best texture image criteria besides its pixel value. In this study, a method is proposed to obtain the size of the best texture image for GNBC by nearest neighbor (NN) interpolation optimization. The best texture image size with interpolated pixel values makes GNBC able to distinguish texture images in each class with the highest performance. The first step of the proposed method was to determine the texture image size for training through a combination of row and column sizes in the optimization process. The next important step in generating the new texture images was resizing each of the original texture images using NN interpolation. The next step was to build GNBC based on the new image from interpolation and determine the classification accuracy. The last step was to select the best texture image size based on the largest classification accuracy value as the first criterion and image size as the second criterion. The evaluation of the proposed method was carried out using texture image data from the CVonline public dataset involving several test scenarios and interpolation methods. The test result shows that in scenarios involving five classes of texture images, GNBC with NN interpolation gives the smallest classification accuracy value of 89% and the largest 100% at the best image size, 14 × 32 and 47 × 42, respectively. In scenarios involving small to large class numbers, GNBC with NN interpolation provides classification accuracy of 81.6%–95%. From these results, GNBC with NN optimization gives better results than other nonadaptive interpolation methods (bilinear, bicubic, and Lanczos) and principal component analysis (PCA).

KEYWORDS — Image, Texture, Interpolation, Naïve Bayes, Nonadaptive, Accuracy.

I. INTRODUCTION

One of the criteria of a classification method's performance in classifying image objects is to yield high accuracy in classification. The high accuracy yielded by the classification method is influenced by several factors, including the accuracy in taking object image samples, image preprocessing, image feature extraction, and the classification method's reliability. The commonly performed image preprocessing is the image scaling and image quality improvement with certain filters. Meanwhile, for image feature extraction, the Fourier transform [1], wavelet, and principal component analysis (PCA) [2], [3] are often used. Image scaling is usually carried out by resizing the original image so that the size (dimensions) of the image for training and testing becomes identical. The resized image size then becomes the input of one or more conventional classifiers, such as naïve Bayes. However, for different image sizes, classifiers may provide different classification performances. The performance of such classifiers may be better or worse. Therefore, getting the right image size to yield the best classifier performance is an important step in the classification process.

There is a limited amount of research that has specifically examined the use of image resizing to improve classification performance. Most researchers solely focus on image improvement using one or more methods of image interpolation [4]–[8]. In image classification, image resizing (e.g., image texture) presents a different challenge to the success of the classification method. The facts of the study demonstrate that researchers seldom employ interpolation to determine the optimal image size for a classification method, and even fewer studies utilize interpolation results to determine the feature value of an image object.

Several previous studies have focused on improving the classification method, including by modifying the model structure or by improving the parameter estimation (weights) of the model. One relatively simple and often used probability-based classification method is the Gaussian naïve Bayes classifier (GNBC). Several GNBC-related studies focusing on method improvement have modified the model structure [9] and improved the parameter estimation (weights) of the model [10]–[14]. Among all these studies, it is known that, in principle, GNBC is a classification method that still assumes that one attribute (feature) is independent of another attribute (feature) [15]–[17]. The assumption makes the GNBC method easy to use or implement; therefore, it has always been a point of discussion and remains the subject of ongoing research, especially regarding how to improve classification performance.

Considering the drawbacks and advantages of the GNBC method and the potential use of interpolation methods, this study proposes a method to obtain the best texture image size using interpolation and to improve the GNBC performance in texture image classification. The contribution of this study is the utilization of the interpolation method to obtain the best texture image size on GNBC, improvement of the GNBC performance with the best texture image size using interpolation nearest neighbor (NN) and reducing the texture image size for input on GNBC.

II. METHODOLOGY

In broad outline, the stages in this research include acquiring texture images for training, optimizing image size by employing the NN interpolation method against GNBC performance, and obtaining the best (optimal) texture image

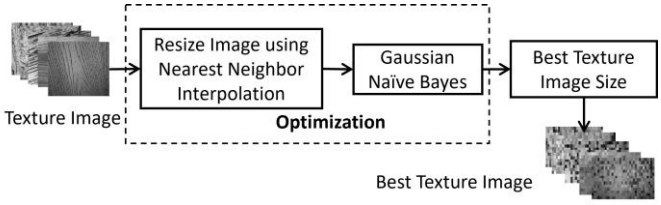


Figure 1. Main process of the Nearest Neighbor interpolation optimization to get the best texture image.

size for feature extraction, as illustrated in Figure 1. Image acquisition is executed to retrieve texture image data per class, obtain the pixel value of each image, and collect the value in the form of a matrix [18]. In this research, the interpolation method was employed to determine the pixel values of the new size image with respect to the original image. A new image size was created, beginning with the smallest size and gradually increasing, with a combination of rows and columns, to the largest size. In addition, this was not created based on the ratio to the original image. Principally, image size optimization with NN interpolation is to make each new resized texture image of the original texture image as training input to GNBC. Next, a process was carried out to obtain the best texture image size among the new images that had been formed. The best texture image appropriate for GBBC was obtained from the best texture image size in such a way that the GNBC could distinguish the texture image of each class with the best performance.

A. NEAREST NEIGHBOR-GAUSSIAN NAÏVE BAYES CLASSIFIER (NN-GNBC)

Interpolation by NN is the simplest interpolation. As a result, the computational time of the method is relatively faster than other interpolation methods. This condition occurs because of the principle of the interpolation method, which involves selecting the pixel value in its surrounding area. The new pixel value is obtained from the nearest pixel value to minimize or enlarge the original image [4]. The kernel from the NN interpolation is mathematically written as follows:

$$l(x) = \begin{cases} 0 & |x| \geq 0.5 \\ 1 & |x| < 0.5 \end{cases} \quad (1)$$

where $|x|$ is the distance between the point and the grid of points to be interpolated on the texture image. The kernel function was utilized to obtain the pixel value of the interpolated result.

If it is known that $T = \{z_1, \ldots, z_d\}$ is the image resulting from the interpolation with NN and is considered as an attribute value (feature) which includes $z_1, \ldots, z_d$, where d is the image feature size (row × column), then the image feature is then used to create or build GNBC [19]. In the creation of the GNBC, it was assumed that, among the attributes (features), the image was independent, and it was also assumed that each feature or attribute had a normal distribution (Gaussian). Furthermore, taking into account these assumptions, the probability value of the attribute (feature), if the - j th (C_j) class is known, is as follows [17].

$$P(T|C_j) = \prod_{k=1}^{d} P(T_k|C_j) = \prod_{k=1}^{d} N(z_k; \hat{\mu}_{jk}, \hat{\sigma}_{jk}) \quad (2)$$

where $\hat{\mu}_{jk}$ and $\hat{\sigma}_{jk}$ are the estimated values of the mean and standard deviation parameters for the kth attribute and jth class. The following equation was derived through the utilization of the conditional probability rule.

$$P(C_j, T) = P(C_j)P(T|C_j) = P(C_j)\prod_{k=1}^{d} N(z_k; \hat{\mu}_{jk}, \hat{\sigma}_{jk}). \quad (3)$$

From (3), further classification results could be determined based on the attributes (features) of the inputted image and the greatest probability value of $P(C_j, T)$ in the following equation.

$$\hat{c} = \underset{c_j}{argmax}\{P(C_j)\prod_{k=1}^{d} N(z_k; \hat{\mu}_{jk}, \hat{\sigma}_{jk})\} \quad (4)$$

where $\hat{c}$ is the classification result.

To address the computational problem, logarithmic operations could be performed on (3), and it yielded discriminant functions for each class, as follows.

$$g_j(T) = log(P(C_j)) - \sum_{k=1}^{d} log(\hat{\sigma}_{jk}) - \frac{1}{2}\sum_{k=1}^{d} \frac{(z_k - \hat{\mu}_{jk})^2}{\hat{\sigma}_{jk}^2} \quad (5)$$

where g_j is the discriminant of the jth class.

Equations (1) to (5) demonstrate the connection between the NN interpolation method and GNBC, resulting in the term NN-GNBC. In this case, the GNBC performance level was strongly influenced by the interpolated results. In addition, not all interpolation results yielded satisfactory performance. Therefore, to obtain interpolation results yielding a high-performance level on GNBC, it was necessary to optimize the interpolation method. While the image yielded through NN interpolation might not be as high in quality as other interpolation methods, it did not necessarily mean that the interpolation negatively impacts GNBC's ability to classify image objects.

B. NEAREST NEIGHBOR INTERPOLATION OPTIMIZATION ALGORITHM

The NN interpolation optimization in this study is resizing all images in training to obtain optimal image size. The optimization was carried out to determine the right texture image size and interpolation value for GNBC so as to distinguish the images in each class with the highest accuracy. The first step to achieve the best image size was to modify the size of all texture images for training into a new texture using NN interpolation according to the predetermined image size. Second, the GNBC was built based on the input of new texture images resulting from interpolation. Third, all interpolated new texture images used for training were classified. Then, the first to third steps were iterated with various combinations of row sizes and texture image columns. Furthermore, the GNBC model, which achieved the highest accuracy while utilizing the smallest texture image size, was selected. Next, the fifth step was to obtain the texture image size with the highest GNBC accuracy as the best texture image, as shown in the Optimization-NN-GNBC Algorithm.

The inputs to the Optimization-NN-GNBC Algorithm are rowIm, colIm, nSample, Im, and C. The rowIm input is the row size of the training image, while colIm is the size of the training image column, assuming each training image size is identical. The nSample input is the number of images used in training, Im is the original image, and C is the class or label in each training image. Meanwhile, the algorithm output is bestSizeIm, which shows the best training image size and provides the highest GNBC classification accuracy. As explained earlier, in this research, the training image was resized to various possible sizes. In this algorithm, image resizing was carried out starting from the smallest row and column size of the training image (1 × 1) to the largest size (rowIm × colIm). The resizeIm is the result of image resizing for each size, which is obtained by the

NN interpolation function using the NN interpolation kernel as in (1). Next, the result of each resizing and those applied to each training image were converted into an (X) vector with the reshape function. The accuracy value in the classification for each image size was obtained by utilizing the TrainingGNBC function. Then, each result of image resizing and its accuracy was stored in rowdim, coldim, and accuracyGNBC. From the optimization results, followed by utilizing the OptimalImageDim function, the best image row and column size with the highest GNBC accuracy was obtained and accommodated in bestSizeIm. The best image row and column size is the size with the smallest row and column multiplication with the highest GNBC accuracy.

Algorithm of Optimization-NN-GNBC
//Optimization of NN interpolation to get the best image size for GNBC
Input: rowlm, collm, nSample, Im, C
//rowIm: the size of image row, colIm: the size of image column, nSample: the number of //image in training, Im: original image and C: class or label
Output: bestSizeIm
//image row and column size that produces the best accuracy for GNBC
iter ← 0 *//initialization*
for i = 1 to rowIm do
  for j = 1 to colIm do
    for k = 1 to nSample do
      resizeIm ← interpolasiNN(Im(k),i,j)
      //resize image with NN kernel using (1)
      X(k,1: i*j) ← reshape (resizeIm,1,i*j)
    end
    accuracy ← TrainingGNBC(X,C)
    //using (4) and (5)
    iter ← iter+1
    rowdim(iter) ← i
    coldim(iter) ← j
    accuracyGNBC(iter) ←accuracy
  end
end
bestSizeIm ← OptimalImageDim(rowdim, coldim, accuracyGNBC)

The bestSizeIm variable was obtained using the OptimalImageDim function which process was carried out using the Optimal-Image-Dim algorithm. In principle, the algorithm runs by finding the highest GNBC accuracy value (bestaccuracy) from various sizes of training images resulting from resizing using interpolation. However, the highest accuracy value may be obtained on several image sizes. As a result, if there are several of the same highest accuracy values, the utilized criterion is the smallest resized image size. The accuracyGNBC (i) = bestaccuracy operation was performed to check for the same high accuracy value. In addition, to obtain the smallest image size, the bindex function was employed, i.e., the multiplication index of the row size by the image column size (bsize), which value is the smallest. Furthermore, from these indices, the most optimal image size was obtained (bestrowsize, bestcolsize, and bestaccuracy).

Algorithm of Optimal-Image-Dim
//Function to obtain the best image size
Input: rowdim, coldim, accuracyGNBC
Output: bestrowsize, bestcolsize, bestaccuracy
//bestrowsize: the best size of row, bestcolsize: the best size of image column
//bestaccuracy: the best accuracy for GNBC
j ← 0 *//inizialitation*
bestaccuracy ← max(accuracyGNBC)
for i = 1 to rowdim*coldim do
  if (accuracyGNBC(i) = bestaccuracy)
    j ← j +1
    browsize(j) ← rowdim(i)
    bcolsize(j) ← coldim(i)
    bsize(j) ← browsize(j)*bcolsize(j)
  end
end
bindex ← min(bsize)
bestrowsize ←browsize(bindex)
bestcolsize ← bcolsize(bindex)

C. THE BEST FEATURE SELECTION ALGORITHM FROM NN INTERPOLATION OPTIMIZATION RESULTS

If the best image size and pixel values obtained from NN interpolation optimization are considered to represent the main features of the image, then the value of the image feature extraction results is equivalent to the value of the image interpolation results on the best image size. The extraction value of the feature is then accommodated in the vector, as shown in the Algorithm of Feature-NN. The feature extraction value significantly impacts the structure of the created GNBC model, as well as the results of model parameter estimation and classification. In the Algorithm of Feature-NN, the feature extraction value is bestresize, obtained based on the best image size (bestrowsize and bestcolsize), and the interpolasiNN function. The image interpolation result was then converted into the *y* vector.

Algorithm of Feature-NN
//Feature extraction for GNBC based on results of NN interpolation optimization
Input: bestrowsize, bestcolsize, Im
Output: y
//the best image feature for GNBC from NN optimization results
bestresize ← interpolasiNN (Im, bestrowsize, bestcolsize)
y (1: bestrowsize*bestcolsize) ← reshape (bestresize, 1, bestrowsize* bestcolsize)

III. TEST SCENARIO

Multiple tests with several scenarios were conducted in this study to evaluate the performance of the proposed method. The tests were conducted using public data provided by Lazebnik, Schmid, and Ponce, which is accessible at di CVonline: Image Databases in the form of texture image datasets [20]. The dataset consists of 25 classes/labels; each class contains 40 texture image samples, so the total number of samples is a thousand texture images (grayscale). Each image in the dataset has the same resolution, 480 × 640 pixels. As shown in Table I and Table II, several groups of datasets and test scenarios are prepared based on the image dataset.

Scenario 1 uses the dataset in Table I involving five classes/labels, including texture images T01-T05, texture images T06-T10, texture images T11-T15, texture images T16-T20, and texture images T21-T25. NN interpolation optimization is performed in each dataset scenario to obtain the optimal training image size for GNBC. The test in scenario 1 aims to find out the performance of the proposed method more deeply by looking at the value of accuracy, precision, sensitivity (recall), and F-score [21], [22].

The next scenario, i.e., scenario 2, involves the dataset in Table II. The test process is almost the same as the test in scenario 1, with the exception that the data in scenario 2 is used

TABLE I
DATASET DESCRIPTION SCENARIO 1: TEXTURE IMAGE (FIVE CLASSES)

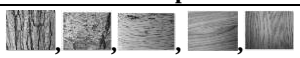
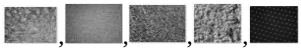
Image	Class	Example
T01-T05	bark1-bark2- bark3-wood1- wood2	
T06-T10	wood3- water-granite- marble-floor1	
T11-T15	floor2-pebbles-wall-brick1-brick2	
T16-T20	glass1-glass2-carpet1-carpet2-uphostery	
T21-T25	wallpaper-fur-knit-corduroy-plaid	

TABLE II
DATASET DESCRIPTION OF SCENARIO 2: TEXTURE IMAGE (2 TO 25 CLASSES)

Texture Image		Multiple Classes		Sample Size	
T01-T02	T01-T11	2	11	80	440
T01-T03	T01-T13	3	13	120	520
T01-T05	T01-T17	5	17	200	680
T01-T07	T01-T21	7	21	280	840
T01-T09	T01-T25	9	25	360	1,000

to determine the consistency of the proposed method's performance. Therefore, in this scenario, the training image dataset was piloted with a small number of classes (two classes/labels), namely (T01-T02), to the training image with large classes (25 classes/label), namely (T01-T25).

To determine the success rate of NN interpolation optimization on GNBC performance, the results of this optimization were also compared with the results of optimization using other nonadaptive interpolation methods, namely bilinear, bicubic, and Lanczos, as well as optimization using the PCA method [3], [23]. The bilinear and bicubic methods in previous studies yielded sharper images than NN, but the computation time was greater than NN [24]. In general, in the bicubic method, the value of a is usually determined from the value of -0.5 to -0.75 [25]–[27]. In this study, the value of a = -0.5 was used. For the Lanczos interpolation method [6], [8], [28], [29], in this study, the value of a is determined to be positive integer 2 (a = 2), also called the 2nd order of Lanczos interpolation.

In order to facilitate the comparison of the methods tested in each scenario, several treatment methods were named as follows.

- The process of resizing texture images using NN interpolation to achieve the best training image size for GNBC is referred to as NN-GNBC.
- The process of resizing texture images using bilinear interpolation to achieve the best training image size for GNBC is referred to as BL-GNBC.
- The process of resizing texture images using bicubic interpolation to achieve the best training image size for GNBC is referred to as BC-GNBC.
- The process of resizing texture images using the 2nd order of Lanczos interpolation (a=2) to achieve the best training image size for GNBC is referred to as LZ-GNBC.
- The process of resizing texture images using NN interpolation, followed by feature extraction using PCA based on the selected eigenvalue (λ) for GNBC, is referred to as NN-PCA-GNBC.

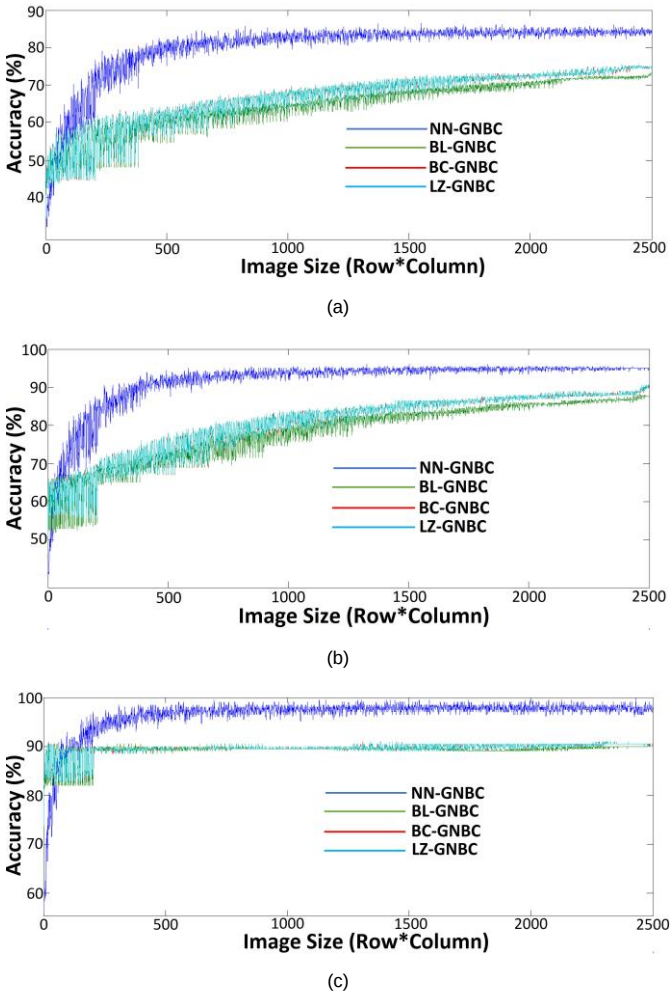


Figure 2. Example of the optimization process to obtain the best image dimensions/size for GNBC on scenario 1, (a) T01-T05, (b) T06-T10, (c) T16-T20.

IV. RESULTS AND DISCUSSION

A. SCENARIO 1

The test in scenario 1 yielded the following results. First, it was the results of the optimization process for resizing the texture image using an interpolation method based on the accuracy value of GNBC classification. Then, the second result was the best texture image size obtained from the optimization process. The third test result was the outcome of evaluating GNBC performance according to the best input texture image. It considered several factors, such as accuracy, precision, sensitivity, and F-score values. The test results in each scenario were limited to the maximum size of image rows and columns in an iteration, which were 50 and 50 (50 × 50). Therefore, the number of combinations of texture image size in the optimization process was 2,500. The iteration process was carried out to obtain the best texture image. The best criteria were based on the best classification accuracy value as the first priority and the smallest image size as the next priority.

The iteration process in optimization to obtain the best image size using the T01-T05 texture image dataset in scenario 1 is shown in Figure 2(a). According to the figure, it is evident that when the size (row × column) < 15, the NN-GNBC tends to yield smaller accuracy values compared to other methods. However, for images with a size (row × column) ≥ 15, the NN-GNBC yields better accuracy values than BL-GNBC, BC-GNBC, and LZ-GNBC. The BC-GNBC and LZ-GNBC

TABLE III
SIZE OF THE BEST IMAGE OPTIMIZATION RESULTS IN SCENARIOS 1 AND 2

Image Texture Dataset		λ PCA*	The Best Texture Image Size			
			NN-GNBC	*BL-GNBC*	*BC-GNBC*	*LZ-GNBC*
Scenario 1	T01-T05	$\lambda_{1,..,36}$	14 × 32	47 × 49	41 × 50	41 × 50
	T06-T10	$\lambda_{1,..,38}$	20 × 27	40 × 50	46 × 48	45 × 48
	T11-T15	$\lambda_{1,..,39}$	47 × 8	46 × 50	40 × 48	40 × 48
	T16-T20	$\lambda_{1,..,37}$	47 × 42	4 × 5	3 × 3	22 × 25
	T21-T25	$\lambda_{1,..,36}$	4 × 35	27 × 36	24 × 19	23 × 20
Scenario 2	T01-T02	$\lambda_{1,..,36}$	9 × 17	2 × 2	17 × 4	44 × 6
	T01-T03	$\lambda_{1,..,36}$	8 × 21	37 × 9	14 × 11	47 × 6
	T01-T05	$\lambda_{1,..,36}$	14 × 32	47 × 49	41 × 50	41 × 50
	T01-T07	$\lambda_{1,..,38}$	30 × 19	47 × 49	41 × 50	41 × 50
	T01-T09	$\lambda_{1,..,38}$	41 × 17	49 × 50	41 × 50	45 × 40
	T01-T11	$\lambda_{1,..,38}$	30 × 19	43 × 48	41 × 50	41 × 50
	T01-T13	$\lambda_{1,..,39}$	26 × 27	49 × 50	50 × 50	50 × 50
	T01-T17	$\lambda_{1,..,37}$	26 × 35	50 × 50	45 × 50	45 × 50
	T01-T21	$\lambda_{1,..,39}$	26 × 46	50 × 50	50 × 50	45 × 50
	T01-T25	$\lambda_{1,..,37}$	34 × 36	49 × 50	46 × 50	49 × 50

*) the image used for PCA is 10% of the original size

TABLE IV
GNBC ACCURACY OF INTERPOLATION METHOD OPTIMIZATION RESULT ON SCENARIOS 1 AND 2

Texture Image Dataset		Accuracy (%)				
		NN-PCA-GNBC	*NN-GNBC*	*BL-GNBC*	*BC-GNBC*	*LZ-GNBC*
Scenario 1	T01-T05	63.00	89.00	78.50	79.00	79.00
	T06-T10	76.50	96.00	88.00	90.50	90.50
	T11-T15	49.00	93.00	84.50	86.00	86.00
	T16-T20	69.50	100	91.00	91.00	91.00
	T21-T25	62.50	94.00	89.00	89.00	89.50
Scenario 2	T01-T02	81.25	95.00	92.50	92.50	93.75
	T01-T03	58.33	93.33	89.17	88.33	89.17
	T01-T05	63.00	89.00	78.50	79.00	79.00
	T01-T07	58.93	88.57	72.86	73.93	73.93
	T01-T09	57.50	86.67	73.33	75.56	75.56
	T01-T11	51.14	85.00	71.14	73.64	73.64
	T01-T13	46.92	86.15	72.50	75.00	75.00
	T01-T17	41.32	82.65	67.21	69.71	69.71
	T01-T21	36.31	82.02	69.88	71.79	71.79
	T01-T25	28.60	81.60	67.20	68.80	68.90

*) the image used for PCA is 10% of the original size

methods yield relatively similar results and are still better than the BL-GNBC method. Then, for the T06-T10 dataset, Figure 2(b) demonstrates that NN-GNBC generally achieves lower accuracy compared to other methods for size (row × column) < 21. However, it performs better than other methods for sizes (row × column) ≥ 21. Similarly, in Figure 2(c), the NN-GNBC model achieves lower accuracy compared to other models for the T16-T20 dataset when the size (row × column) < 41. However, when the size (row × column) ≥ 41, it outperforms other models. Meanwhile, for the T11-T15 and T21-T25 datasets, the optimization process yields almost similar results as in the previous dataset, i.e., NN-GNBC, which obtained better accuracy than BL-GNBC, BC-GNBC, and LZ-GNBC, when the image size (row × column) ≥ 49 and the image size (row × column) ≥ 57.

Based on the optimization process of the training image size on the scenario 1 data, with T1-T5, T06-T10, T11-T15, T16-T20, and T21-T25 image datasets, the highest classification accuracy value along with the training image size is obtained, as shown in Table III and Table IV. The test results show that, overall, NN-GNBC has better accuracy in texture image classification compared to BL-GNBC, BC-GNBC, and LZ-GNBC, with the best (optimal) image size varying for each dataset. In the test with the T01-T05 dataset, the optimization results show that the best NN-GNBC is generated at an image size of 14 × 32 with 89% accuracy, while the best BL-GNBC is at an image size of 47 × 49 with 78.5% accuracy, the best BC-GNBC is at an image size of 41 × 50 with 79% accuracy, and the best LZ-GNBC is at an image size of 41 × 50 with 79% accuracy. According to the findings, it is evident that NN-GNBC yields the best result in terms of accuracy and image size (the smallest). Generally, the same results were obtained for experiments T06-T10, T11-T15, and T21-T25. The NN-GNBC performed the best in terms of accuracy or image size (the smallest). However, in experiments with the T16-T20 dataset, although the best image size obtained by NN-GNBC remained bigger than others, its accuracy value remained the highest.

TABLE V
EXAMPLE OF NN INTERPOLATION RESULTS ON IMAGES T01-T05

Texture Image	Original Image (480 × 640)	NN Interpolation Result Image (14 × 32)
T01		
T02		
T03		
T04		
T05		

In order to compare with another feature extraction method, specifically the PCA method, the test began by downsizing the original image from 480 × 640 pixels to 48 × 64 using NN interpolation. The test only employed NN because NN gave the best performance on GNBC among other interpolation methods. Image feature extraction with PCA was performed with several eigenvalues. The utilized eigenvalues, along with their best accuracy, are shown in Table III and Table IV. The test results show that feature extraction using PCA with GNBC (NN-PCA-GNBC) yields worse results than other methods.

In general, the test results in scenario 1 show that NN-GNBC yields the best accuracy compared to other methods. These results cannot be separated from the success of NN interpolation in transforming the original image. With its kernel function, NN can interpolate new image pixel values of different sizes, making it easier for GNBC to distinguish one texture image from another. Table V presents visual examples of the results obtained through interpolation using NN.

The performance level of each method in scenario 1 can be seen from the precision, recall, and F-score values in Table VI. Test results with the T01-T05 image dataset show that NN-GNBC generally has a much better precision, recall, and F-score (T01, T02, T03, T04, T05) than BL-GNBC, BC-GNBC, and LZ-GNBC, although for T03 image, all methods yield the same precision value. All test results on the dataset show that the performance of NN-GNBC in classifying texture images is generally better than BL-GNBC, BC-GNBC, and LZ-GNBC. BC-GNBC obtains precision, recall, and F-score, which is relatively the same as LZ-GNBC, while BL-GNBC yields the lowest performance.

TABLE VI
PRECISION, RECALL, AND F-SCORE OF TEST RESULTS ON SCENARIO 1

Image	Precision (%)				Recall (%)				F-Score (%)			
	NN-GNBC	*BL-GNBC*	*BC-GNBC*	*LZ-GNBC*	*NN-GNBC*	*BL-GNBC*	*BC-GNBC*	*LZ-GNBC*	*NN-GNBC*	*BL-GNBC*	*BC-GNBC*	*LZ-GNBC*
T01	97.06	85.71	85.71	85.71	82.50	75.00	75.00	75.00	89.19	80.00	80.00	80.00
T02	80.00	73.08	73.58	73.58	100	95.00	97.50	97.50	88.89	82.61	83.87	83.87
T03	100	100	100	100	87.50	85.00	85.00	85.00	93.33	91.89	91.89	91.89
T04	82.61	68.29	68.29	68.29	95.00	70.00	70.00	70.00	88.37	69.14	69.14	69.14
T05	91.43	71.05	72.97	72.97	80.00	67.50	67.50	67.50	85.33	69.23	70.13	70.13
T06	100	97.22	100	100	97.50	87.50	90.00	90.00	98.73	92.11	94.74	94.74
T07	100	95.00	95.12	95.12	100	95.00	97.50	97.50	100	95.00	96.30	96.30
T08	100	77.55	82.22	82.22	100	95.00	92.50	92.50	100	85.39	87.06	87.06
T09	100	92.11	97.22	97.22	82.50	87.50	87.50	87.50	90.41	89.74	92.11	92.11
T10	83.33	81.08	80.95	80.95	100	75.00	85.00	85.00	90.91	77.92	82.93	82.93
T11	100	93.94	97.06	97.06	87.50	77.50	82.50	82.50	93.33	84.93	89.19	89.19
T12	85.11	86.36	88.37	88.37	100	95.00	95.00	95.00	91.95	90.48	91.57	91.57
T13	100	97.44	100	100	100	95.00	95.00	95.00	100	96.20	97.44	97.44
T14	90.91	64.91	65.52	65.52	100	92.50	95.00	95.00	95.24	76.29	77.55	77.55
T15	91.18	92.59	92.59	92.59	77.50	62.50	62.50	62.50	83.78	74.63	74.63	74.63
T16	100	88.24	86.11	87.88	100	75.00	77.50	72.50	100	81.08	81.58	79.45
T17	100	90.00	87.80	94.74	100	90.00	90.00	90.00	100	90.00	88.89	92.31
T18	100	100	100	100	100	97.50	97.50	97.50	100	98.73	98.73	98.73
T19	100	78.72	81.82	76.00	100	92.50	90.00	95.00	100	85.06	85.71	84.44
T20	100	100	100	100	100	100	100	100	100	100	100	100
T21	91.67	78.95	79.49	79.49	82.50	75.00	77.50	77.50	86.84	76.92	78.48	78.48
T22	95.00	88.37	90.24	90.48	95.00	95.00	92.50	95.00	95.00	91.57	91.36	92.68
T23	84.09	78.57	78.57	80.49	92.50	82.50	82.50	82.50	88.10	80.49	80.49	81.48
T24	100	100	100	100	100	92.50	92.50	92.50	100	96.10	96.10	96.10
T25	100	100	97.56	97.56	100	100	100	100	100	100	98.77	98.77

For testing on T06-T10 images, the results show that besides yielding the best accuracy, NN-GNBC also yields precision, recall, and F-score (T06, T07, T08, T09, T10) which are relatively higher than BL-GNBC, BC-GNBC, and LZ-GNBC. However, for image T09, NN-GNBC has a smaller recall than BL-GNBC, BC-GNBC, and LZ-GNBC. Thus, in comparison to BL-GNBC, BC-GNBC, and LZ-GNBC, NN-GNBC yields a better texture image classification for the experiment on T06–T10 images. In the test with the dataset, BC-GNBC obtains precision, recall, and F-score relatively similar to LZ-GNBC, while BL-GNBC yields the lowest performance.

For the experiments on T11-T15 images, besides producing the best accuracy, in general, NN-GNBC also has precision, recall, and F-score (T11, T12, T13, T14, T15), which are relatively better than BL-GNBC, BC-GNBC, and LZ-GNBC. However, for T12 and T15, the precision is smaller than the others. Therefore, the NN-GNBC test on T11-T15 images generally yields better classification performance than the others.

Experiments with T16-T20 and T21-T25 images also showed the same results. NN-GNBC obtained higher values of precision, recall, and F-score (T16, T17, T18, T19, T20), which are relatively higher than BL-GNBC, BC-GNBC, and LZ-GNBC. According to the test results on the dataset, it has been found that NN-GNBC outperforms BL-GNBC, BC-GNBC, and LZ-GNBC in classifying texture images.

B. SCENARIO 2

Tests in scenario 2 were used to determine the consistency of the proposed method's performance on various utilized sample sizes, datasets, and classes/labels. For the test using T01-T02 texture images (two classes), the optimization process to obtain the best texture image size is shown in Figure 3. From Figure 3(a), it can be seen that at an image size (rows × columns) < 518, NN-GNBC generally yields lower classification accuracy relative to other methods. However, for a texture image size (row × column) ≥ 518, NN-GNBC provides the best accuracy value compared to BL-GNBC, BC-GNBC, and LZ-GNBC. The optimization process on T01-T13 images representing medium classes (13 classes) and images T01-T025 representing many classes (25 classes) is shown in Figure 3. From Figure 3(b), it can be seen that with the T01-T13 dataset, NN-GNBC at image size (row × column) < 25 also generally yields relatively lower accuracy values than other methods, but at different image sizes, it yields the best accuracy values. BC-GNBC and LZ-GNBC gave relatively similar results, while BL-GNBC gave relatively lower accuracy than all methods tested. On T01-T025 datasets, NN-GNBC on image size (row × column) ≥ 31 obtains the best accuracy value compared to other methods, as shown in Figure 3(c). Meanwhile, tests with texture images of T01-T09 (nine classes), T01-T17 (17 classes), and T01-T21 (21 classes) yield relatively similar results to those with other datasets.

The optimization process on dataset scenario 2, i.e., T01-T02, T01-T03, T01-T05, T01-T07, T01-T09, T01-T11, T01-T13, T01-T17, T01-T21, and T01-T25, results in the best training image size and the best classification accuracy value as shown in Table III and Table IV. The test results show that, overall, NN-GNBC yields better texture image classification accuracy than other methods. The best image size is different for each training image dataset. For example, for the experiments on texture images T01-T02, the optimization results show that NN-GNBC is optimal (best) at an image size

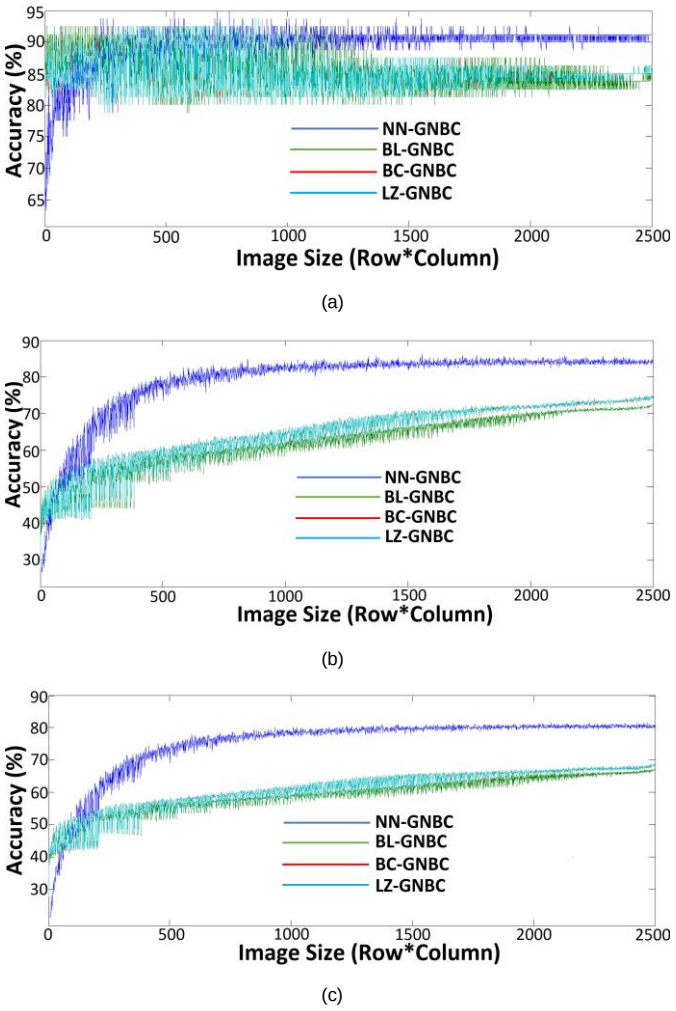


Figure 3. Optimization process to get the best texture image dimensions/size for GNBC on scenario 2, (a) T01-T02, (b) T01-T13, (c) T01-T25.

of 9 × 17 (153) with 95% accuracy. In comparison, BL-GNBC is optimal at 2 × 2 (4) with 92.5% accuracy, BC-GNBC at 17 × 4 (68) with 92.5% accuracy, and LZ-GNBC at 44 × 6 (264) with 93.75% accuracy. In these tests, the NN-GNBC method was the best in accuracy, although not in image size. Meanwhile, for experiments on all image datasets, NN-GNBC got the best results in accuracy and image size. Although for experiments T01-T03, NN-GNBC is not the best image size, the resulting accuracy value is still the best. For comparing other feature extraction methods, namely the PCA method, the test results show that feature extraction with PCA (NN-PCA-GNBC) obtained worse accuracy than other methods on all dataset images.

If the tests on the dataset of T01-T02 (two classes), T01-T03 (three classes), and T01-T05 (five classes) images represent tests with few classes, then T01-T07 (seven classes), T01-T09 (nine classes), T01-T11 (eleven classes), and T01-T13 (13 classes) represent tests with a medium number of classes, and T01-T17 (17 classes), T01-T21 (21 classes), and T01-T25 (25 classes) represent tests with many classes, then the overall test results show that NN-GNBC is more consistent (stable) with the best texture image classification accuracy. In addition, the result of training image size with NN-GNBC is relatively better than that of other methods.

Despite the success of NN-GNBC in obtaining the best results in resizing texture images in the scenarios conducted, it is realized that there are some limitations in this experiment, including the need to involve several feature extraction methods from image objects and several classification methods in the experiment, so that more general conclusions can be drawn.

V. CONCLUSION

Optimization of the NN interpolation method has significantly contributed to obtaining the best texture image size against GNBC. The contribution level of the NN interpolation method to improve GNBC performance based on the accuracy, precision, sensitivity, and F-score values in classifying texture images can be determined by optimizing the NN interpolation method. Experimental results using several dataset scenarios show that texture image resizing based on optimizing NN interpolation yields better accuracy values than the use of bilinear, bicubic, and Lanczos interpolation and better than PCA. On the other hand, the optimization of texture image interpolation using bicubic and Lanczos results in relatively equal and better performance than interpolation bilinear. Experimental results with several datasets of texture images that are considered to represent small, medium, and large class sizes also show that the NN interpolation method more consistently provides the best texture image results with a classification accuracy level of GNBC better than other interpolation methods. Determining the best image size optimized using the interpolation method is expected to be an alternative for image reduction and feature extraction from image objects before the classification process.

CONFLICTS OF INTEREST

The authors declare no conflict of interest in the article "The Best Texture Image for Gaussian Naïve Bayes With Nearest Neighbor Interpolation."

AUTHORS' CONTRIBUTIONS

Conceptualization, Irwan Budi Santoso and Shoffin Nahwa Utama; methodology, Irwan Budi Santoso; software, Irwan Budi Santoso; validation, Irwan Budi Santoso, Shoffin Nahwa Utama, and Supriyono; formal analysis, Irwan Budi Santoso; investigation, Irwan Budi Santoso; data curation, Irwan Budi Santoso; writing-original draft, Irwan Budi Santoso; writing-reviewing and editing, Shoffin Nahwa Utama and Supriyono; visualization, Irwan Budi Santoso.

ACKNOWLEDGMENT

The researchers are grateful for the support in putting this research into practice provided by the scientific group team of the Informatics Engineering study program at Universitas Islam Negeri Maulana Malik Ibrahim in Indonesia.

REFERENCES

[1] M.S. Nixon and A.S. Aguado, *Feature Extraction and Image Processing*, 1st ed. Oxford, England: Newnes, 2002.

[2] Y.-H. Shin, M.-J. Park, O.-Y. Lee, and J.-O. Kim, "Deep orthogonal transform feature for image denoising," *IEEE Access*, vol. 8, pp. 66898–66909, Apr. 2020, doi: 10.1109/ACCESS.2020.2986827.

[3] I.T. Jolliffe, *Principal Component Analysis*, 2nd ed. New York, NY, USA: Springer-Verlag, 2002.

[4] S.H. Mahajan and V.K. Harpale, "Adaptive and non-adaptive image interpolation techniques," *2015 Int. Conf. Comput. Commun. Control Automat.*, 2015, pp. 772–775, doi: 10.1109/ICCUBEA.2015.154.

[5] G. Ramesh and T.A. Prasath, "An aphoristic study on different interpolation techniques for medical image scaling and its comparative analysis," *2021 Int. Conf. Comput. Commun. Inform. (ICCCI)*, 2021, pp. 1–4, doi: 10.1109/ICCCI50826.2021.9402675.

[6] D. újica-Vargas, Y. Mújica-Vargas, M.M. Cruz, and A. Rendón-Castro, "Improvement of MRI images through heterogeneous interpolation techniques," *2019 Int. Conf. Electron. Commun. Comput. (CONIELECOMP)*, 2019, pp. 112–117, doi: 10.1109/CONIELECOMP.2019.8673230.

[7] P. Bhatt, S. Patel, and R. Pandit, "Comparative analysis of interpolation and texture synthesis method for enhancing image," *Int. J. Innov. Res. Sci. Eng. Technol.*, vol. 2, no.1, pp. 278–283, Jan. 2013.

[8] S. Safinaz and A.V.R. Kumar, "VLSI realization of Lanczos interpolation for a generic video scaling algorithm," *2017 Int. Conf. Recent Adv. Electron. Commun. Technol. (ICRAECT)*, 2017, pp. 17–23, doi: 10.1109/ICRAECT.2017.37.

[9] I.B. Santoso, Supriyono, C. Crysdian, and K.F.H. Holle, "Optimization of naïve Bayes classifier to classify green open space object based on Google Earth image," *2018 Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, 2018, pp. 465–469, doi: 10.1109/ISRITI.2018.8864279.

[10] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Mach. Learn.*, vol. 29, pp. 103–130, Nov. 1997, doi: 10.1023/A:1007413511361.

[11] G.N. Norén and R. Orre, "Case based imprecision estimates for Bayes classifiers with the Bayesian bootstrap," *Mach. Learn.*, vol. 58, pp. 79–94, Jan. 2005, doi: 10.1007/s10994-005-5010-y.

[12] M. Ekdahl and T. Koski, "Bounds for the loss in probability of correct classification under model based approximation," *J. Mach. Learn. Res.*, vol. 7, pp. 2449–2480, Nov. 2006.

[13] M. Hall, "A decision tree-based attribute weighting filter for naive Bayes," *Knowl.-Based Syst.*, vol. 20, no. 2, pp. 120–126, Mar. 2007, doi: 10.1016/j.knosys.2006.11.008.

[14] T.-T. Wong, "Alternative prior assumptions for improving the performance of naïve Bayesian classifiers," *Data Min. Knowl. Discov.*, vol. 18, no. 2, pp. 183–213, Apr. 2009, doi: 10.1007/s10618-008-0101-6.

[15] G. Shobha and S. Rangaswamy, "Machine learning," in *Handbook of Statistics 48: Deep Learning*, V. Gavindaraju, A.S.R.S. Rao, and C.R. Rao, Eds., Cambridge, USA: Academic Press Publications, 2018, pp. 197–228, doi: 10.1016/bs.host.2018.07.004.

[16] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inf. Syst*., vol. 14, no. 1, pp. 1–37, Jan. 2008, doi: 10.1007/s10115-007-0114-2.

[17] A.R. Webb and K.D. Copsey, *Statistical Pattern Recognition*, 3rd ed. Hoboken, USA: John Wiley & Sons, Ltd., 2011.

[18] R.C. Gonzalez, R.E. Woods, and S.L. Eddins, *Digital Image Processing Using MATLAB*, 2nd ed. Knoxville, USA: Gatesmark Publishing, 2009.

[19] S. Iqbal *et al.*, "Prostate cancer detection using deep learning and traditional techniques," *IEEE Access*, vol. 9, pp. 27085–27100, Feb. 2021, doi: 10.1109/ACCESS.2021.3057654.

[20] S. Lazebnik, C. Schmid, and J. Ponce (2003) The texture database, CVonline image database. [Online], http://www-cvr.ai.uiuc.edu/ponce_grp/data/#texture, access date: 18-Apr-2017.

[21] D.M.W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," 2020, *arXiv.2010.16061*.

[22] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.

[23] W. Zhao, Y. Lv, Q. Liu, and B. Qin, "Detail-preserving image denoising via adaptive clustering and progressive PCA thresholding," *IEEE Access*, vol. 6, pp. 6303–6315, Dec. 2017, doi: 10.1109/ACCESS.2017.2780985.

[24] T. Acharya and P.-S. Tsai, "Computational foundations of image interpolation algorithms," *Ubiquity*, vol. 2007, no. October, pp. 1–17, Oct. 2007, doi: 10.1145/1322464.1317488.

[25] Z. Xingyu, W. Yong, and L. Xiaofei, "Approach for ISAR imaging of near-field targets based on coordinate conversion and image interpolation," *J. Syst. Eng. Electron.*, vol. 32, no. 2, pp. 425–436, Apr. 2021, doi: 10.23919/JSEE.2021.000036.

[26] K.-L. Chung, C.-Y. Huang, and C.-W. Kao, "An effective bicubic convolution interpolation-based iterative luma optimization for enhancing quality in chroma subsampling," *IEEE Access*, vol. 9, pp. 149744–149755, Nov. 2021, doi: 10.1109/ACCESS.2021.3125713.

[27] C.-S. Tsai, H.-H. Liu, and M.-C. Tsai, "Design of a scan converter using the cubic convolution interpolation with Canny edge detection," *2011 Int. Conf. Elect. Inf. Control Eng.*, 2011, pp. 5813–5816, doi: 10.1109/ICEICE.2011.5777979.

[28] R.V. Sharan and T.J. Moir, "Time-frequency image resizing using interpolation for scoustic event recognition with convolutional neural networks," *2019 IEEE Int. Conf. Signals Syst. (ICSigSys)*, 2019, pp. 8–11, doi: 10.1109/ICSIGSYS.2019.8811088.

[29] A. Puziy, I. Gavrilov, K. Nosirov, and A. Akhmedova, "Efficiency estimation of image resizing based on interpolating transformations," *2019 Int. Conf. Inf. Sci. Commun. Technol. (ICISCT)*, 2019, pp. 1–5, doi: 10.1109/ICISCT47635.2019.9011885.