The Visual Computer
https://doi.org/10.1007/s00371-022-02572-5

ORIGINAL ARTICLE

Optimal feature selection and classification of Indian classical dance hand gesture dataset

R. Jisha Raj[1] · Smitha Dharan[1] · T. T. Sunil[2]

Accepted: 2 May 2022

Abstract
Classical dancers all over the world use various body gestures to communicate to the audience the intended meaning. The study of these gestures can help in better understanding of the dance forms and also for annotation purposes. Bharatanatyam, an Indian classical dance, uses elegant hand gestures (mudras), facial expressions and body movements to convey the meaning to the audience. There are 28 Asamyukta Hastas (single-hand gestures) and 23 Samyukta Hastas (double-hand gestures) in Bharatanatyam. Open datasets on Bharatanatyam dance gestures are not presently available. An exhaustive open dataset of 15,396 distinct single-hand gesture images and 13,035 distinct double-hand gesture images was created. In this paper, we intend to find an optimal feature descriptor for this dataset. Various feature descriptors like scale-invariant feature transform (SIFT), speeded-up robust features (SURF), oriented FAST and rotated BRIEF (ORB), KAZE, KAZE extended, accelerated-KAZE (A-KAZE), binary robust invariant scalable keypoints (BRISK) and SIKA were explored. The feature descriptors were coded using bag of visual words and classified using several classifiers like support vector machines (SVM), multilayer perceptron (MLP), Naive Bayes, logistic regression, decision tree, AdaBoost and random forest. From all these investigations, it is observed that KAZE descriptor and random forest combination has the topmost classification accuracy.

Keywords Bag of visual words · Bharatanatyam mudra dataset · Ensemble classifiers · Multilayer perceptron · Random forest · SIFT

1 Introduction

India is a culturally rich country where different genres of artistic expressions exist. Dance forms in India have a rich and vital tradition that dates back to ancient times [57]. Excavations, literary sources and paintings of different periods give extensive evidence of the practice of dance.

Classical dancers connect with the audience through their abhinaya (mudras and facial expressions) and whole-body movements. They transport the meaning of an accompanying "shloka" or poetry played in the background. Bharatanatyam is a popular Indian classical dance form from South India [57]. This traditional dance form comprises elegant and graceful hand movements (mudras) which are expressive and meaningful [50]. According to Natyashastra, a classical text on Indian dance, by Bharata Muni, there are 28 Asamyukta Hastas (single-hand gestures) and 23 Samyukta Hastas (double-hand gestures) [15,19,47] in Bharatanatyam.

Bharatanatyam dance involves several intricate movements which are to be studied for years under the expert supervision of gurus or teachers. The absence of experts is one of the reasons for the decline of this art form. Another reason is the difficulty in comprehending the meaning of mudras, facial expressions and body movements of the dancer. These two reasons have hindered the reachability of Bharatanatyam to new learners and dance enthusiasts all over the world. If this state continues, the decline of this dance form is inevitable.

Computer vision techniques can contribute a lot toward the revival of this ancient art form. This work is the first step toward realizing an automatic dance gesture identification.

✉ R. Jisha Raj
jisharaj@ceconline.edu

Smitha Dharan
smitha@ceconline.edu

T. T. Sunil
vu2swx@gmail.com

1 College of Engineering (Affiliated to APJ Abdul Kalam Technological University, Kerala), Chengannur 689121, Kerala, India

2 College of Engineering, Attingal 695101, Kerala, India

Published online: 27 June 2022

Automatic dance gesture identification has several applications like developing an e-learning tool that can assist in self-study of the dance, annotating dance videos and providing a live commentary during a stage performance. These assistive interventions will be a large step toward the revival of the dance form. These innovative techniques can also benefit other dance forms in India and other parts of the world.

At present, there is an unavailability of open datasets on Bharatanatyam dance gestures. An exhaustive dataset[1] comprising of various mudras in Bharatanatyam was created. Our dataset has 15,396 distinct single-hand gesture images and 13,035 distinct double-hand gesture images.

In this paper, a comparative study of the performance of seven different feature descriptors and seven different classifiers have been made to find out the most accurate feature descriptor–classifier combination for building the mudra classifier. The performance of the following feature descriptors is evaluated: (1) SIFT (scale-invariant feature transform) [31], (2) SURF (speeded-up robust features) [6], (3) ORB (oriented fast and rotated brief) [49], (4) KAZE [1] and KAZE extended, (5) A-KAZE (accelerated-KAZE) [2], (6) BRISK (binary robust invariable scalable keypoints) [27] and (7) SIKA [54] (SIFT+KAZE) [54].

All these descriptors work in conjunction with bag of visual words (BoVW) method. The BoVW vector is classified using various classifier algorithms as follows: (1) support vector machines (SVMs), (2) multilayer perceptron (MLP), (3) Naive Bayes, (4) decision tree, (5) logistic regression, (6) AdaBoost and (7) random forest. The performance of various feature descriptor and classifier combinations in classifying the Bharatanatyam mudra dataset is evaluated in terms of classification accuracy and area under ROC curve (ROC-AUC score).

2 Literature review

Feature descriptors identify points in an image that can be used to describe the image's distinct features such as edges, corners, ridges and blobs [51]. All classifiers irrespective of their application comprise feature extraction and classification sections. Choosing the right features improves the classifier accuracy.

A comparison of the feature descriptors has been made by authors for various applications like image matching, image retrieval, image classification, face identification, etc. Krystian et al. have compared the performance of several descriptors for image matching [34]. In [32], six feature descriptors were compared on their performance of face identification in videos. Vassileious et al. developed HPatches, a benchmark for the evaluation of feature descriptors [5].

SIFT (scale-invariant feature transform) [31] is one of the most powerful descriptors available. Ever since the origin of SIFT descriptors, they have been extensively used by researchers in image classification. SIFT descriptors have been used in the field of remote sensing [58], action recognition [52], leaf image classification [56], Batik fabric image classification [4] and also in large-scale image classification [53]. Speeded-up robust features (SURF) [6] was experimented upon by many authors for various applications such as face antispoofing [11], video stabilization [44], visual tracking in real scenarios [28] and object matching [55]. KAZE feature descriptor [1] was used in visual odometry [14], hyper-spectral image alignment [38], offline signature verification [37], etc. A-KAZE features [2] were applied by various authors for vehicle classification [7], image matching [22,30] and tracking [43]. BRISK algorithms [27] have been experimented upon in fields like target identification [29], image matching [23,59], etc. Siddharth et al. developed a new feature descriptor, SIKA (combining SIFT and KAZE feature descriptors) followed by an SVM classifier [54].

In the context of Bharatanatyam mudras, studies based on very limited datasets were made by few authors. Mozarkar et al. [36] used 68 samples of 13 static double-hand gestures of Bharatanatyam to develop a recognition system and Sriparna Saha et al. [50] used 28 single-hand gestures of Bharatanatyam to develop such a system. A rotation- and scale-invariant gesture recognition system which could be applied for Bharatanatyam dance mudra recognition was experimented upon by Divya et al. [21]. A classifier for pose and hand gesture classification developed by Aparna Mohanty et al. [35] used 1400 samples of 10 single-hand gestures and 840 samples of 14 double-hand gestures of Bharatanatyam. Anami et al. used 2400 images of 24 double-hand gesture images and used vertical and horizontal cross sections of mudra contour as a feature vector and classified using a rule-based classifier [3]. Anuja et al. used 27 classes of single-hand gesture images of Bharatanatyam collected from Google images and YouTube videos. After data augmentation, there were 18,992 images which were classified by CNN models that used the principle of transfer learning [39]. Kishore et al. used HOG features and SVM classifier to classify 120 images of single- and double-hand mudras [26]. Srimani et al. classified 230 images of 23 double-hand gestures using orientation histogram and skeletal-based matching [25].

In this work, feature descriptors such as SIFT, SURF, KAZE, KAZE extended, A-KAZE, BRISK and SIKA algorithms work in conjunction with the bag of visual words (BoVW) model. Seven classifier algorithms were used to classify the BoVW vector.

[1] Full dataset can be downloaded from https://github.com/jisharajr/Bharatanatyam-Mudra-Dataset.git.

3 Data acquisition and dataset description

According to Natyashastra by Bharata Muni, there are 28 single-hand gestures (Asamyukta Hastas) and 23 double-hand gestures (Samyukta Hastas) in Bharatanatyam. With the help of 15 volunteers who were trained in Bharatanatyam for more than 5 years, 15,396 single-hand gesture images and 13,035 double-hand gesture images were collected. One of the single-hand mudras being dynamic was excluded. *Katakamukha mudra* has three variations, and each of them was taken as a separate class. This made up to 29 classes of static single-hand mudras in the dataset. Due to the inclusion of a large part of the dancer's body in the captured images, two mudras in the double hand were excluded from the dataset. Thus, double-hand mudras in the dataset are comprised of 21 different classes.

The dataset includes different views of the same mudra as perceived by a spectator. As the dancer moves on stage, a spectator sees the mudras at different angles. Considering this, 50–60 views of each mudra were obtained in succession. But some images were redundant due to close similarities [24]. The similarity of images, captured from a single volunteer, was calculated using mean square error (MSE) and structural similarity index measurement (SSIM) as in Eqs. (1) and (2) [24]. If I and K are two images of the same dimension $m \times n$,

$$MSE = 1/mn \sum_{i=1}^{n} \sum_{j=1}^{m} (I(i,j) - K(i,j))^2 \quad (1)$$

Images with mean square error value zero are the same images, and dissimilar images have values greater than zero. This value grows as dissimilarity increases. Structural similarity index [13] attempts to model the structural information of images.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_1)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (2)$$

SSIM compares images on a block by block basis. The parameters of Eq. (2) are as follows: (x, y) indicates the location of $N \times N$ block in each image; μ_x, μ_y are mean intensities in x, y directions; σ_x, σ_y is the variance of intensities in x, y directions; and σ_{xy} is the covariance of the block. SSIM value is between -1 and $+1$. Value of 1 indicates perfect similarity [13]. For each mudra from a particular volunteer, MSE and SSIM were calculated between successive images. Only those images with an MSE value greater than 40 and SSIM value lesser than 0.92 were considered distinct. Thus, 30–45 distinct views of a mudra from a volunteer were included in the dataset. As an example, six different views of *Hamsapaksha mudra* are shown in Fig. 1.

The images were captured in a studio environment using Apple iPhone 6S 12 megapixel camera. The original images were in RGB format of size 3000 × 4000. The camera was mounted on a tripod stand and kept at one meter distance from screen. A green screen served as the permanent background. The data collection environment is shown in Fig. 2.

The flowchart of the experiment design is shown in Fig. 3.

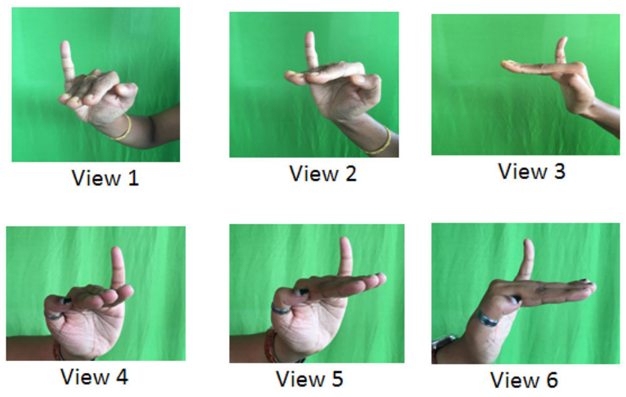
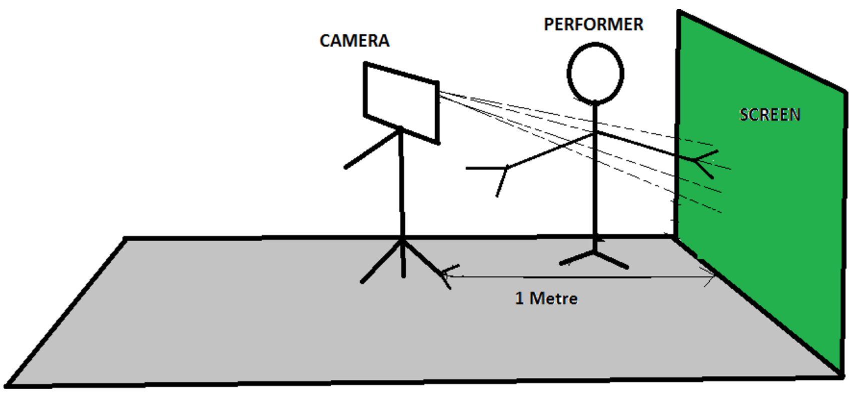
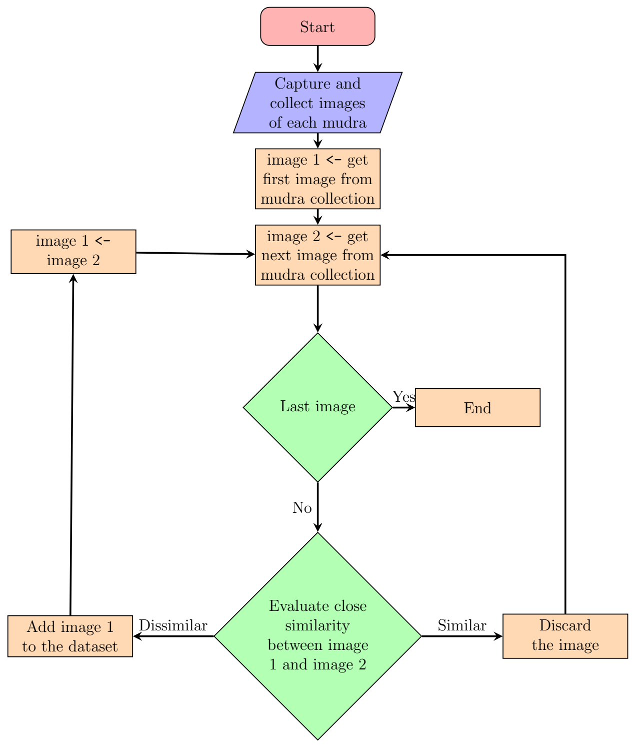


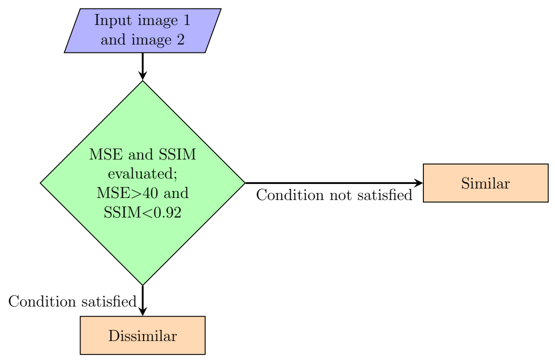
Fig. 1 Six different views of *Hamsapaksha mudra*, one of the single-hand mudras

Fig. 2 Data collection environment




Start
Capture and collect images of each mudra
image 1 <- get first image from mudra collection
image 2 <- get next image from mudra collection
image 1 <- image 2
Last image
Yes
End
No
Evaluate close similarity between image 1 and image 2
Dissimilar
Add image 1 to the dataset
Similar
Discard the image

(a)


Input image 1 and image 2
MSE and SSIM evaluated; MSE>40 and SSIM<0.92
Condition not satisfied
Similar
Condition satisfied
Dissimilar

(b)

Fig. 3 **a** Flowchart for image acquisition. **b** Flowchart of similarity comparison between images

4 Mudra classifier

A classification system comprising three components was developed. The block diagram of the proposed mudra classifier is shown in Fig. 4. The main components are as follows:

1. Feature detector and descriptor
2. Bag of visual words
3. Classifiers.

In this paper, an attempt was made to compare various feature detector and descriptor algorithms such as SIFT, SURF, ORB, KAZE, KAZE Extended, A-KAZE, BRISK and SIKA in context of Bharatanatyam mudra dataset.

All images were first resized to 100 × 100 and grayscale-converted. These images were fed to the feature extraction block which provided the feature descriptors. The feature descriptors were encoded using bag of visual words and then classified using various algorithms like support vector machines (SVM), multilayer perceptron (MLP), Naive Bayes, decision tree, logistic regression, AdaBoost and random forest. An attempt is made to find the best feature descriptor–classifier combination that will lead to a highly accurate classification system for the Bharatanatyam mudra dataset.

4.1 Feature detector and descriptor

Image feature detection consists of keypoint detection. Keypoints are "points of interest" in an image. They are spatial locations or points in the image that define what is interesting or what stands out in an image. Keypoints will have descriptors attached with them. These descriptors are concerned with the scale and orientations of the keypoints. Keypoint selection is done so that the features selected give the best classifier performance according to common evaluation matrices such as precision, recall, F1-score, accuracy and AUC-ROC score [54]. This removes data redundancy and contributes to computational efficiency. Figure 5 shows the keypoints of various feature descriptors on a single-hand mudra image.

4.1.1 SIFT algorithm

Scale-invariant feature transform (SIFT) [31] was proposed by David Lowe in 1999. The keypoints are invariant to scale variations and rotations [20]. SIFT is also robust to noise, changes in illumination and a small number of viewpoint changes. To obtain scale-invariant features, SIFT uses a multiresolution pyramid over the input image. For this scale space generation, Laplacian of Gaussian (LoG) is used. Difference of Gaussian (DoG) is used for approximation of LoG. A point qualifies as a keypoint if it is the maximum or minimum considering other points in a 3 × 3 × 3 neighborhood in scale space. For keypoint orientation, a 16 × 16 neighborhood around a keypoint is considered. It is then divided into 4 × 4 subregions, and the orientation of the pixels in it is quantized to eight major directions [31]. Thus, SIFT feature descriptor is a 128 length vector.

4.1.2 SURF algorithm

Speeded-up robust features (SURF) algorithm is a local feature detector and descriptor which is computationally faster and more robust than SIFT [6,20]. While SIFT used DoG

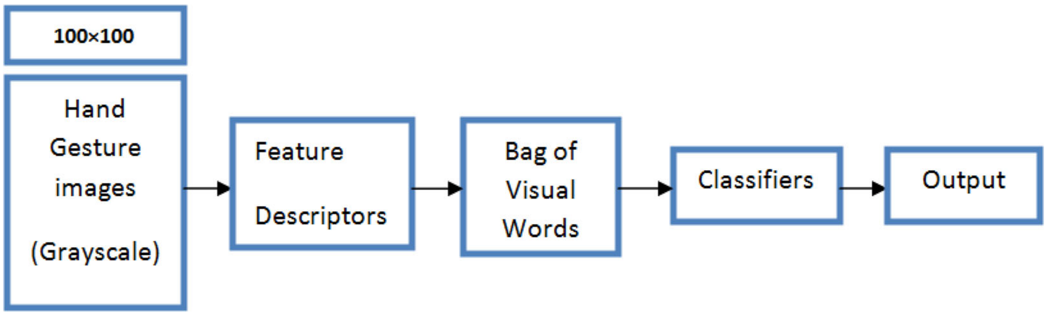


Fig. 4 Block diagram of the proposed mudra classifier

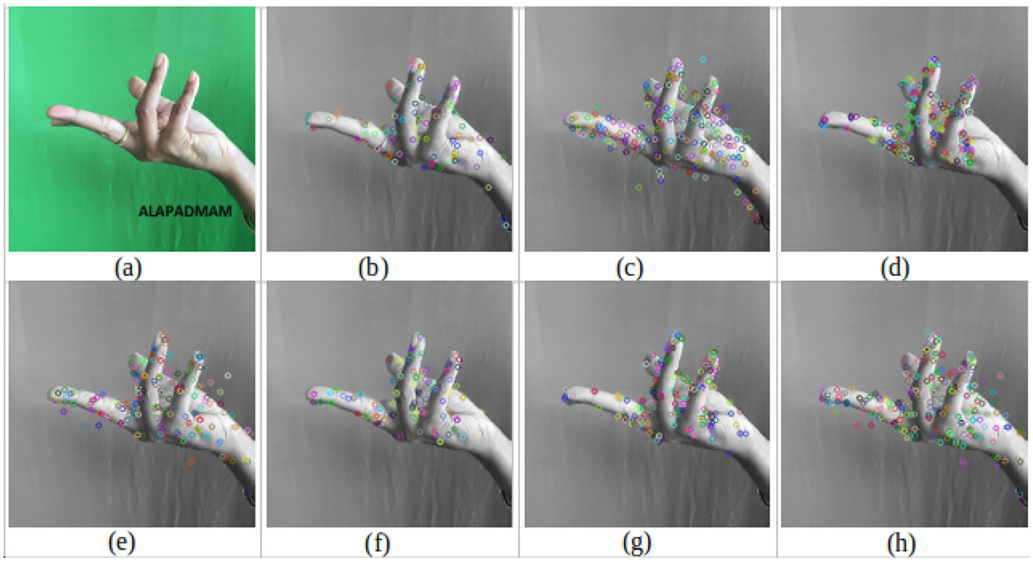


Fig. 5 Keypoints of various descriptors. **a** Original single-hand mudra image, **b** SIFT, **c** SURF, **d** ORB, **e** KAZE, **f** A-KAZE, **g** BRISK, **h** SIKA

(difference Of Gaussian) to approximate LoG (Laplacian of Gaussian) for scale space generation, SURF used box filter approximation to achieve this. Convolution using box filter is very easily achieved when integral images are used. And this can be done in parallel for different scales. Determinant of Hessian matrix (DoH) is used for interest point detection.

For being invariant to rotation, reproducible orientation needs to be identified for the keypoints. SURF uses Haar wavelet response in horizontal and vertical directions for a circular neighborhood of defined size centered at the keypoint [6]. The circular region is of radius 6s where s is the scale at which interest point is identified. The wavelet responses are represented as vectors in a space with horizontal response strength along the x-axis and vertical response strength along the y-axis [6].

A sum of horizontal and vertical responses in a sliding orientation window of 60^0 is obtained. The two sums yield a vector. The direction of the longest such vector is identified as the direction of keypoint [6].

For extraction of feature descriptor, first a square region of size 20s is centered around the interest point and is made to orient along the same direction as obtained in the previous step. The square region is divided into 4×4 subregions [6]. Haar responses for the horizontal and vertical directions, d_x and d_y, are obtained for each subregion. These responses are summed, and a four-dimensional descriptor vector is obtained for each subregion $v = \{\Sigma d_x, \Sigma d_y, \Sigma|d_x|, \Sigma|d_y|\}$. Thus, considering all the 4×4 subregions the descriptor will be of length 64 [6].

4.1.3 ORB algorithm

Oriented FAST and rotated BRIEF (ORB) is an efficient and viable alternative to SIFT or SURF [49]. It uses the FAST algorithm as a feature detector and rotated BRIEF as binary descriptor. FAST (features from accelerated segment test) is a template-based corner detection scheme used to determine the interest points. FAST uses an intensity threshold between the center pixel and those in a circular ring about the center. It does not provide multiscale features. To achieve this, ORB uses the FAST algorithm repeatedly to each layer of scale pyramid of the image. FAST has a high response at edges, and hence a Harris corner filter is used to reject FAST keypoints at edges. ORB uses FAST-9 which performs well [17,49].

BRIEF is a binary feature descriptor. Once a keypoint is detected, a neighborhood around keypoint is defined. This is called a patch. BRIEF converts these patches into feature vectors by pairwise intensity comparisons. The feature vectors so generated are variant to rotation. To address this issue, ORB uses a rotation-aware variant of BRIEF [49]. For each patch, a centroid is obtained and the direction of the vector from keypoint (patch center) to patch centroid determines keypoint's direction. The binary test pattern is then rotated by the direction of a patch before binary descriptor extraction, allowing a feature to be represented in a rotation-invariant (i.e., isotropic) form [49].

4.1.4 KAZE algorithm

KAZE is a feature detector and descriptor [1] which operates in the nonlinear scale space contrary to SIFT or SURF which find features in the Gaussian scale space (particular instance of linear diffusion). However, Gaussian blurring smoothens out boundaries and noise to the same degree when evolving the original image through the scale space [1].

In nonlinear scale space details are preserved while removing noise. In KAZE, nonlinear filtering along with a conductance function is used instead of Gaussian kernel as in SIFT [1]. The nonlinear filtering equation is

$$\frac{\partial I(x, y, t)}{\partial t} = div[c(x, y, t) \cdot \nabla I(x, y)] \tag{3}$$

where *div* and ∇ are divergence and gradient operators. c is the conductivity function and I is the intensity image. In nonlinear filtering, image is filtered iteratively by preserving edges, provided the conductivity function is chosen carefully. A good conductivity function [1] is

$$c(x, y, t) = \left[1 + \left\{\frac{\nabla I_\sigma(x, y, t)}{k}\right\}^2\right]^{-1} \tag{4}$$

where I_σ is the Gaussian smoothened version of image I. While SIFT uses DoG to build scale space, KAZE uses an additive operator splitting (AOS) scheme. Determinant of Hessian matrix (DoH) is used to detect a keypoint.

For orientation computation, a neighborhood around the keypoint is considered and the gradient of each pixel is obtained. These gradients are represented in vector space, and the longest vector with dominant orientation is assigned as the orientation of the keypoint [1]. Modified SURF descriptors which can work in nonlinear spaces are used here.

4.1.5 A-KAZE algorithm

Accelerated-KAZE works in the same way as that of KAZE but uses a faster method to create the nonlinear space called fast explicit diffusion (FED) [2]. In addition to the use of FEDs, A-KAZE uses a binary descriptor to further increase the speed. It makes use of modified version of local difference binary (LDB) descriptor. LDB follows the same principle as BRIEF but computes the binary test only in a region around the keypoint [2]. A-KAZE is faster than KAZE and obtains comparable results to KAZE in some datasets.

4.1.6 BRISK algorithm

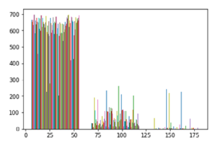
Binary robust-invariant scalable keypoints (BRISK) is a low computational feature detector scheme [27]. Here, interest point detection is performed using AGAST (adaptive and generic corner detection based on accelerated segment test). It is an extension of FAST algorithm. Here, scale space is created using octaves and intra-octaves. Each octave is half-sampled from previous ones, and intra-octaves are obtained by down-sampling to be placed between octaves. Non-maximal suppression is applied to each octave and non-octave and the sub-pixel maximum is computed in each patch. Once keypoints are detected, BRISK uses a sampling pattern to sample points around a keypoint [27]. To prevent aliasing effect while sampling a pixel, Gaussian smoothing is applied to each pixel. Point pairs of these pixels are separated into long and short segment pairs. The gradient is computed on long segment pairs to determine the feature orientation.

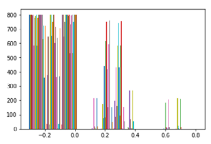
4.1.7 SIKA

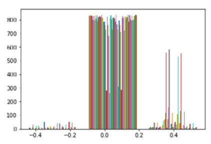
SIKA features [54] complement the strength of SIFT (scale-invariant feature transform) with that of KAZE. SIFT has keypoints in the entire region of the object, and KAZE has keypoints concentrated along the boundary. This is due to the difference in the generation of the keypoints. SIFT uses Gaussian scale space build using DoG (difference of Gaussian), and KAZE uses nonlinear scale space build using nonlinear diffusion filtering. Another difference is that the base image for every octave is obtained from a down-sampled version of the image from the previous octave, whereas for KAZE every octave is generated from the original image. This helps in retaining the boundary points in KAZE [54].

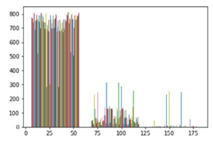
4.2 Bag of visual words

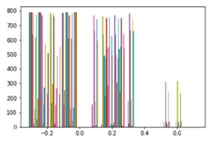
Bag of visual words represents images as a set of features just like a bag of words represents a set of words. In a bag of words, the frequency of occurrence of words in text data is obtained. Here, instead of words, image features are considered.


(a) Visual words histogram of SIFT features from single hand gesture images.


(c) Visual words histogram of SURF features from single hand gesture images.


(e) Visual words histogram of KAZE features from single hand gesture images.


(b) Visual words histogram of SIFT features from double hand gesture images.


(d) Visual words histogram of SURF features from double hand gesture images.

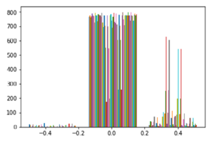

(f) Visual words histogram of KAZE features from double hand gesture images.

Fig. 6 Visual words histograms of some feature descriptors from single- and double-hand gesture images

Image features consist of keypoints and descriptors. These image features are clustered using k-means clustering. The cluster centers will be the code words or visual words of the visual dictionary. Histograms of visual words of some of the feature descriptors extracted from single- and double-hand gesture images are shown in Fig. 6. Next, from each image, features are extracted and mapped to code words through clustering. Each image is represented as a histogram of the code words. Histogram generated for a single- and double-hand gesture image is shown in Fig. 7a, b. The disadvantage of this encoding scheme is its high computational complexity [54].

4.3 Classifiers

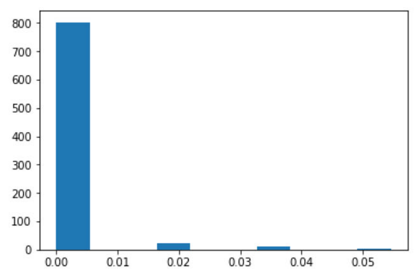
The encoded feature descriptors are classified using different algorithms as follows:

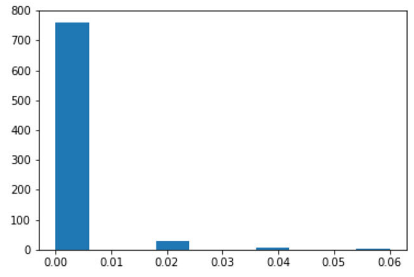
1. Support vector machines
2. Multilayer perceptron
3. Naive Bayes
4. Decision tree
5. Logistic regression
6. Ensemble classifiers.
   (a) AdaBoost algorithm
   (b) Random forest algorithm.

5 Results and discussion

A total of 15,396 single-hand gesture images of 29 classes and 13,035 double-hand gesture images of 21 classes were resized to 100 × 100 and grayscale-converted, and features were extracted. For each descriptor algorithm, the keypoints of all images were stacked and a feature vector was obtained. To encode this vector using bag of visual words method,

Fig. 7 Histogram of code words of a **a** single-hand gesture, *Sikharam mudra* image. **b** Double-hand gesture, *Bherunda mudra* image


(a)


(b)

k-means clustering was used. The cluster size was seen to influence the accuracy of the classifier and so attempt was made with various cluster sizes. The best cluster size in all cases was found to be the square root of the vector. The histograms of images were obtained and were used to train and test seven different classifiers.

The hyperparameters of all classifiers were tuned or optimal values were obtained since these values were dataset dependent. The selection of sub-optimal values which may lead to over-fitting was avoided by using cross-validation procedures for tuning [45].

All classifiers were tuned using GridSearchCV from scikit-learn library (0.24.2) of Python. In GridSearchCV function, fivefold cross-validation was done for tuning the hyperparameters. Once optimal parameters for every classifier were obtained, it was rechecked using a tenfold cross-validation and classification report was generated. Comparative results for various classifiers are shown in Tables 2 and 3. All experiments were done on a machine with Core i9 processor, 32GB RAM, Ubuntu 18.04 LTS and Python 3.6.

5.1 Support vector machines

SVMs are supervised machine learning algorithms that use hyperplanes to perform classification [10,16,41]. They try to maximize the distance between the boundaries demarcating the classes and hence called maximal margin classifiers. By using a kernel function, SVMs map the data points (that are not linearly separable in original space) to a higher-dimensional feature space where they become linearly separable.

Here, trials were performed with linear, polynomial and Gaussian radial basis function (RBF) kernels. One-vs-rest and one-vs-all approaches were explored. Regularization parameter C was kept at 1 in all cases. Polynomial degree was varied from 3 to 10. The accuracy was poor, but among the three kernels, linear showed better performance.

For single-hand gesture images, SVM classifier performed very poorly for all feature descriptors. But for double-hand gestures, SVM showed an accuracy of about 85 and 86% with KAZE and KAZE extended descriptors, respectively. The AUC score was 0.92 for both cases.

5.2 Multilayer perceptron

MLP is a class of neural networks in which the computing elements are arranged in a feed-forward layered structure [12]. It consists of the input layer, hidden layers and output layer. Features in the input decide the number of nodes in the input layer, and the output layer consists of nodes equal to the number of classes. The number of hidden layers and nodes in each hidden layer are estimated on a trial-and-error basis.

For MLP, multiple trials were made by varying the number of hidden layers and by varying the number of neurons in each hidden layer. MLP showed the best performance when the input nodes equaled the cluster size and the output nodes the number of classes. The hidden layers had nodes equal to half of the previous layer. Attempts were made with activation functions like ReLU (rectified linear unit) and sigmoid. The optimizers attempted were Adam and SGD (stochastic gradient descent) with various learning rates like 0.01 and 0.001. ReLU activation function and Adam optimizer with learning rate 0.001 gave the accurate output.

MLP showed about 80% accuracy (AUC score = 0.90) with KAZE and 81% accuracy (AUC = 0.91) with KAZE Extended descriptors for single-hand gestures. For double-hand gestures also, about 94% accuracy (AUC score = 0.97) was obtained with KAZE and KAZE extended descriptors.

5.3 Naive Bayes

Bayesian classifiers assign the most probable class to the example described by feature vectors [9,48]. They are a class of algorithms based on Bayes' theorem. Learning these classifiers is simplified by assuming that features are independent given class.

Here, Gaussian Naive Bayes classifier was used. The only parameter varied for the classifier was variable smoothing. The default value for variable smoothing was e^{-9}. Trials were made with 100 exponential values between 0 and −9. Among these trials, default value gave the best result.

The best performance of this algorithm was with KAZE and KAZE extended descriptors. It showed the accuracy of about 64% (AUC score = 0.81) for single-hand and about 85% (AUC score = 0.91) for double-hand images.

5.4 Logistic regression

Logistic regression uses a sigmoid function to predict the output. The sigmoid function returns a value between 0 and 1. If the function returns a value greater than 0.5, then it is label 1, and if it returns a value lesser than 0.5 then it is label 0. For multiclass classification, one-versus-all approach is used. In this approach, the class which one works with is denoted as 1 and all others zero [9].

In an attempt to fine-tune the classifier, l2 regularization method was chosen and the amount of regularization or C value was varied between values [0.001, 0.01, 0.1, 1, 10, 100]. And optimization algorithms like lbfgs (limited-memory Broyden–Fletcher–Goldfarb–Shanno) and SAG (stochastic average gradient descent) were experimented upon. Although SAG was faster, no further improvement in accuracy was obtained. The optimum values were l2 regularization with C value 1 and lbfgs optimization algorithm.

For single-hand gesture images, it showed poor performance with all descriptors. But for double-hand gesture images, it showed an accuracy of 87% (AUC score = 0.92) with KAZE and KAZE extended.

5.5 Decision tree

The algorithm uses a treelike structure where internal nodes represent decision nodes, branches represent decision rule and leaf nodes represent the outcome. The most important attribute is placed in the root node. One then moves down the tree following the corresponding node that meets our decision. The process finally reaches a leaf node that has the prediction or outcome [8,9].

With KAZE and KAZE extended descriptors, decision tree classifier showed an accuracy of 67% (AUC score = 0.83) for single-hand images and 82% (AUC score = 0.90) for double-hand images.

The impurity criterion of each node was varied between Gini impurity and entropy. Just as stated in the earlier works [46], the tree performance remained the same and so the default, Gini index was chosen. The best split criteria were chosen to split the node since the random split was giving a deeper tree of lesser performance. The depth of the tree was decided by the minimum number of samples at each node. The minimum number of samples at each node could be varied from 1 to 40 [33]. But trials were made by varying the number of samples required to split internal nodes from 2 to 10 . Maximum accuracy was obtained when samples required to split the node was 2.

5.6 AdaBoost

AdaBoost is an ensemble classifier. Here, a sequence of weak learners is connected sequentially so that each model tries to improve and compensate the weakness of its predecessors. Here, the weak learners are decision trees [42].

AdaBoost with fifty decision trees as base estimators and learning rate as 1 were used here and performed poorly with all descriptors. Repeated trials with 100, 300 and 500 estimators were performed, and the learning rate was also varied between 0.001 and 1. But these attempts did not show further improvement from the default values of the algorithm such as 50 estimators and learning rate of 1.

5.7 Random forest

Random forest is also an ensemble classifier with decision trees as base estimators. Here, each decision tree predicts a class and the class that gets the largest number of votes becomes the prediction of the model [42].

The hyperparameters of random forest include the numbers of trees or estimators used to make the decision and the minimum number of samples used by each tree when splitting a node [45]. Here, trials were made with 100, 200 and 300 estimators and the minimum samples at each node was varied from 2 to 10. RF with 100 estimators and minimum samples as 2 showed the best performance.

Random forest was the best-performing classifier showing its topmost accuracy with KAZE and KAZE Extended descriptors. Accuracies of 92% (AUC score = 0.96) and 98% (AUC score = 0.99) were obtained for single- and double-hand gesture images, respectively.

Table 1 shows parameters varied for fine tuning of different classifiers and the optimum parameters chosen based on cross-validation.

Table 1 Parameters varied during fine tuning of classifiers and optimum parameter chosen

Classifier	Parameters varied for different trials	Optimum parameters based on cross-validation
SVM	kernel = linear, polynomial, rbf; multiclass-classification = one-vs-one and one-vs-rest	kernel = linear; multiclass-classification = one-vs-rest
MLP	Activation functions = ReLU, Sigmoid; Optimizers = Adam and SGD; Learning rate = 0.01, 0.001	Activation function = ReLU; Optimizer = Adam; Learning rate = 0.001
Naive Bayes	Variable smoothing = 100 exponential values between 0 and −9.	Variable smoothing $= e^{-9}$ gave best results
Logistic regression	l2 regularization method with amount of regularization or C = [0.001, 0.01, 0.1, 1,10,100]. Optimization algorithms = lbfgs and SAG	l2 regularization method with C = 1 and lbfgs optimization algorithm
Decision tree	Impurity criterion of each node = Gini impurity and entropy; Minimum samples at node for split = 2 to 10	Impurity criterion of each node = Gini impurity; Minimum samples at node for split = 2
AdaBoost	No. of base estimators = 50, 100, 300, 500; learning rate = [0.001, 0.01, 0.1, 1]	No. of base estimators = 50; learning rate = 1
Random forest	No. of estimators = 100, 200, 300; Minimum samples at node for split = 2 to 10	No. of estimators = 100; Minimum samples at node for split = 2

5.8 Summary

KAZE descriptor algorithm along with random forest classifier was found to be the most accurate mudra classifier for both single- and double-hand mudras. Tenfold cross-validated output of four best performing classifiers with KAZE features as input is shown in Fig. 8a, b. KAZE extended descriptor was also implemented and seen to perform with similar accuracy as that of KAZE as shown in Fig. 9a, b. Random forest classifier gave the best accuracy with all the descriptors and is shown in Fig. 10a, b.

The performance of classifiers is evaluated in the paper using five metrics. The accuracy of positive predictions of the

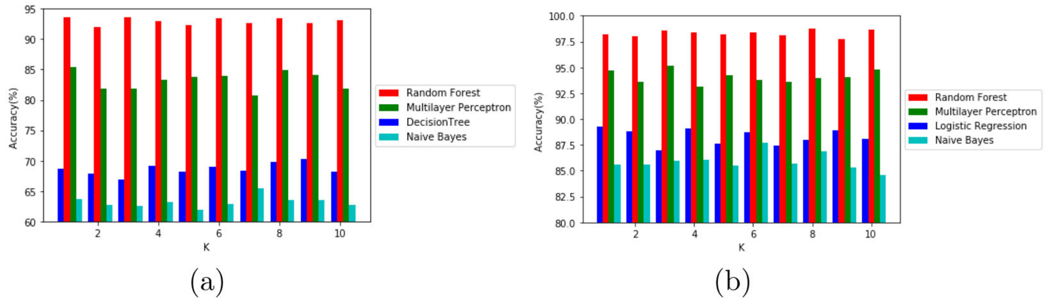


Fig. 8 Tenfold cross-validation of four best-performing classifiers with KAZE features from **a** single-hand gesture images, **b** double-hand gesture images

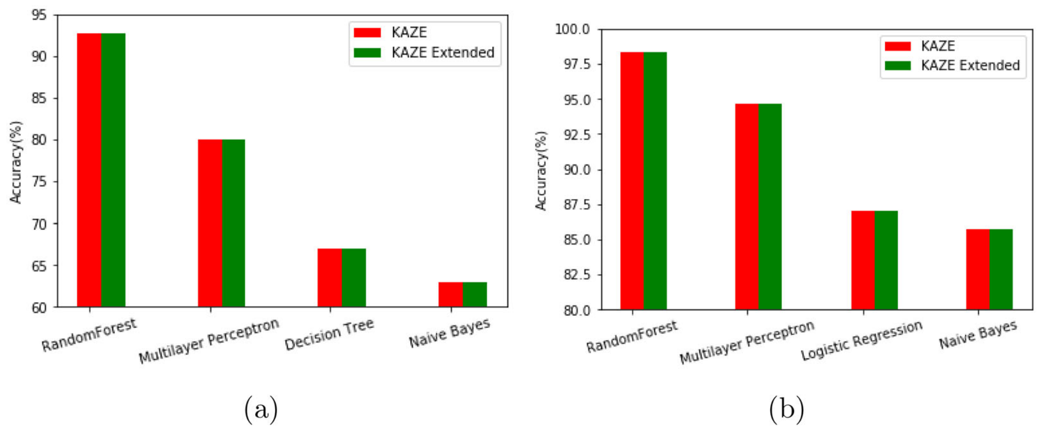


Fig. 9 Four best-performing classifiers with KAZE and KAZE extended features from **a** single-hand gesture images, **b** double-hand gesture images

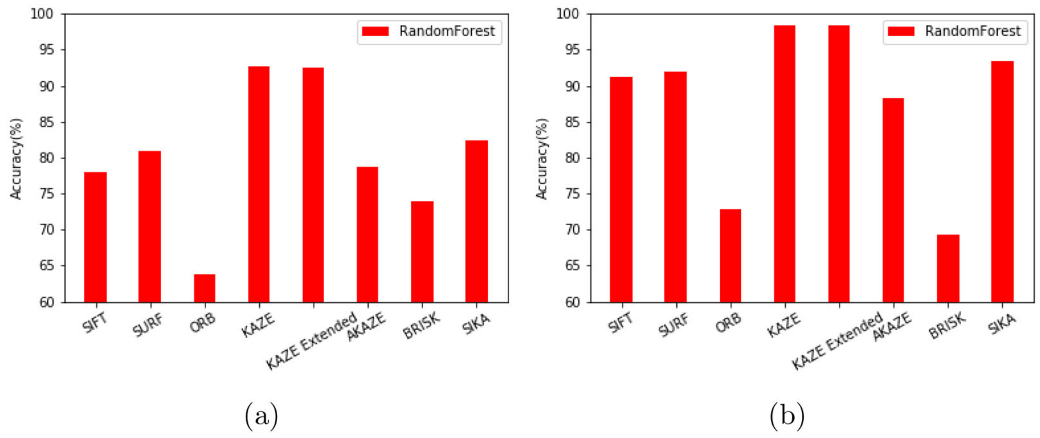


Fig. 10 Random forest classifier accuracy with various feature descriptors extracted from **a** single-hand gesture images, **b** double-hand gesture images

classifier is obtained using a metric called *precision* which is defined as [18,40]

$$precision = \frac{TP}{TP + FP} \tag{5}$$

where *TP* is true positives and *FP* is false positives. The ratio of positive instances correctly detected by a classifier is called *recall* or *sensitivity* or *True Positive Rate (TPR)* which is defined as [18,40]

$$recall = \frac{TP}{TP + FN} \tag{6}$$

where *FN* is false negatives.

Another metric which is used to compare classifiers is F1-score which is a harmonic mean of precision and recall. F1-score will be high only if both precision and recall are high [18,40].

$$F1\text{-}score = 2 * \frac{precision \times recall}{precision + recall} \tag{7}$$

Accuracy is defined as

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{8}$$

where *TN* is true negatives.

Receiver operating characteristics or *ROC* curve plots *recall* vs $(1 - specificity)$. Specificity is the ratio of negative instances correctly classified as negative. Comparison of classifiers is also made by using area under the curve or *AUC*. ROC-AUC score is 1 for a perfect classifier and 0.5 for a purely random classifier [18,40].

Table 2 shows the different feature descriptors, vector obtained by stacking the keypoints of all images and clusters sizes for generating code words using k-means clustering for single-hand mudra images. Precision, recall, F1-score, accuracy and AUC score [40] of various classifiers are also shown. Table 3 summarizes the result when double-hand gesture images are used.

6 Conclusion

This work is a first step toward realizing a fully automatic real time Bharatanatyam mudra classification system. Since open datasets on Bharatanatyam mudras were not available, a new dataset was created. Here, the study is done to find the best feature descriptor for the dataset. Various feature descriptors like SIFT, SURF, ORB, KAZE, KAZE extended, A-KAZE, BRISK and SIKA were attempted. The extracted feature descriptors were k-means clustered and coded using bag of visual words. Here, various cluster sizes were attempted and found that the square root of the feature vector was the most suitable one for all cases. Seven different classifier algorithms were used to classify the BoVW vector. So an exhaustive investigation has been performed, and a comparative study has been made to arrive at an accurate mudra classifier. Various comparisons are summarized in Tables 2 and 3. It is noted that KAZE descriptor along with random forest classifier was the most accurate classifier for both single- and double-hand gesture images. KAZE descriptor gave better performance for all other classifier algorithms also. Thus, the best features from the dataset are extracted by the KAZE descriptor.

The work is compared with other state-of-the-art methods for classification of Bharatanatyam mudras in Table 4. It is observed from the table that our dataset is exhaustive both in terms of the number of classes of mudras and in terms of the number of images collected. The classification accuracy obtained by the proposed method is higher than all works except [35,39]. In [35], the accuracy of the SURF descriptor is 99.8%, but the dataset that the authors used was limited in terms of classes (14 classes of double-hand gestures) and the number of images. In [39], 27 classes of single-hand gesture images were classified using a deep learning approach or convolutional neural network (CNN) classifier, thus yielding an accuracy of 94.5% .

In the future, the possibility of automatic identification of mudras in real time using resource-constrained devices like mobile handsets can also be explored. The speed and memory load of the three most accurate descriptors (KAZE, SIFT and

Table 2 Classification report on using single-hand mudra images

Single-hand mudras												
Descriptors	Vector	Clusters	SVM					MLP				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
(a) Classification report of SVM, MLP, Naive Bayes and logistic regression												
SIFT	(537,848, 128)	700	58%	12%	13%	12%	0.54	62%	60%	60%	60%	0.80
SURF	(645,693, 64)	800	66%	18%	21%	18%	0.57	68%	67%	67%	67%	0.83
ORB	(837,257, 32)	910	20%	10%	15%	18%	0.54	50%	50%	50%	50%	0.72
KAZE	(710,560, 64)	840	37%	37%	37%	37.6%	0.64	80%	80%	80%	80%	0.90
KAZE Extended	(710,560, 128)	840	40%	40%	40%	40.8%	0.65	81%	81%	81%	81%	0.91
A-KAZE	(161,237, 61)	400	48%	48%	49%	48%	0.71	63%	63%	63%	63.7%	0.81
BRISK	(794,254, 64)	900	16%	11%	10%	10%	0.51	53%	53%	54%	54%	0.75
SIKA	(1,248,414, 128)	1120	10%	11%	9%	10%	0.51	62%	61%	60%	61%	0.80

Single-hand mudras												
Descriptors	Vector	Clusters	Naive Bayes					Logistic regression				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
SIFT	(537,848, 128)	700	55%	55%	55%	55%	0.76	46%	46%	46%	46.5%	0.70
SURF	(645,693, 64)	800	57%	57%	57%	57%	0.77	48%	48%	48%	48%	0.71
ORB	(837,257, 32)	910	52%	51%	51%	52%	0.74	32%	32%	32%	31.78%	0.63
KAZE	(710,560, 64)	840	63%	63%	63%	63%	0.81	52%	52%	52%	51.6%	0.74
KAZE Extended	(710,560, 128)	840	65%	65%	65%	64%	0.81	55%	55%	55%	54.6%	0.75
A-KAZE	(161,237, 61)	400	41%	41%	41%	41%	0.68	45%	45%	45%	44.78%	0.70
BRISK	(794,254, 64)	900	60%	60%	60%	60.53%	0.79	43%	43%	43%	43%	0.69
SIKA	(1,248,414, 128)	1120	61%	61%	61%	61%	0.80	11%	11%	11%	11%	0.52

Single-hand mudras												
Descriptors	Vector	Clusters	Decision tree					AdaBoost				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
(b) Classification report of decision tree, AdaBoost and random forest classifiers												
SIFT	(537,848, 128)	700	52%	52%	52%	52%	0.74	14%	14%	14%	14%	0.52
SURF	(645,693, 64)	800	52%	52%	52%	51.8%	0.74	29%	29%	29%	30%	0.62
ORB	(837,257, 32)	910	43%	43%	43%	42.7%	0.69	14%	14%	14%	14%	0.52
KAZE	(710,560, 64)	840	67%	67%	67%	67%	0.83	16%	16%	16%	16%	0.53
KAZE Extended	(710,560, 128)	840	67%	67%	67%	67%	0.83	20%	20%	20%	20%	0.55
A-KAZE	(161,237, 61)	400	60%	60%	60%	60%	0.79	18%	18%	18%	18%	0.54
BRISK	(794,254, 64)	900	53%	53%	53%	52.7%	0.74	27%	27%	27%	27.4%	0.60
SIKA	(1,248,414, 128)	1120	56%	56%	56%	56%	0.77	20%	20%	20%	20%	0.55

Single-hand mudras							
Descriptors	Vector	Clusters	Random forest				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
SIFT	(537,848, 128)	700	78%	78%	78%	78%	0.88
SURF	(645,693, 64)	800	81%	81%	81%	80.96%	0.89
ORB	(837,257, 32)	910	64%	64%	64%	63.78%	0.81
KAZE	(710,560, 64)	840	93%	93%	93%	92.67%	0.96
KAZE Extended	(710,560, 128)	840	92%	92%	92%	92.48%	0.96
A-KAZE	(161,237, 61)	400	79%	79%	79%	78.78%	0.88
BRISK	(794,254, 64)	900	74%	74%	74%	74%	0.85
SIKA	(1,248,414, 128)	1120	82%	82%	82%	82.37%	0.90

Table 3 Classification report on using double-hand mudra images

Double-hand mudras

Descriptors	Vector	Clusters	SVM					MLP				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
(a) Classification report of SVM, MLP, Naive Bayes and logistic regression classifiers												
SIFT	(670,674, 128)	820	44%	44%	44%	44%	0.66	83%	83%	83%	83%	0.90
SURF	(622,695, 64)	790	60%	60%	60%	59.65%	0.80	85%	85%	85%	85%	0.92
ORB	(637,237, 32)	800	41%	41%	41%	41%	0.64	64%	64%	64%	64%	0.80
KAZE	(632,702, 64)	800	87%	86%	85%	85%	0.92	94%	94%	94%	94.39%	0.97
KAZE Extended	(632,702, 128)	800	86%	86%	86%	86%	0.92	94%	94%	94%	94.29%	0.97
A-KAZE	(202,371, 61)	500	85%	85%	85%	84.57%	0.91	83%	83%	83%	82.7%	0.90
BRISK	(598,471, 64)	770	18%	14%	12%	12%	0.51	56%	56%	56%	55.6%	0.76
SIKA	(1,269,426, 128)	1130	12%	14%	15%	12%	0.51	84%	84%	84%	84%	0.91

Double-hand mudras

Descriptors	Vector	Clusters	Naive Bayes					Logistic regression				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
SIFT	(670,674, 128)	820	81%	81%	81%	81.5%	0.90	69%	69%	69%	69.43%	0.84
SURF	(622,695, 64)	790	83%	83%	83%	83%	0.90	75%	75%	75%	75%	0.86
ORB	(637,237, 32)	800	73%	73%	73%	72.72%	0.85	63%	63%	63%	62.78%	0.79
KAZE	(632,702, 64)	800	85%	85%	85%	84.73%	0.91	87%	87%	87%	87%	0.92
KAZE Extended	(632,702, 128)	800	84%	84%	84%	84.11%	0.91	88%	87%	87%	87.21%	0.92
A-KAZE	(202,371, 61)	500	65%	65%	65%	64.74%	0.80	81%	81%	81%	81.39%	0.89
BRISK	(598,471, 64)	770	57%	57%	57%	56.13%	0.77	37%	37%	37%	36.56%	0.66
SIKA	(1,269,426, 128)	1130	83%	83%	83%	83%	0.90	30%	30%	30%	30.43%	0.63

Double-hand mudras

Descriptors	Vector	Clusters	Decision tree					AdaBoost				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC	Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
(b) Classification report of decision tree, AdaBoost and Random forest classifiers												
SIFT	(670,674, 128)	820	60%	60%	60%	60%	0.78	21%	21%	21%	21.16%	0.55
SURF	(622,695, 64)	790	58%	58%	58%	58.11%	0.79	31%	31%	31%	30%	0.62
ORB	(637,237, 32)	800	42%	42%	42%	42.75%	0.70	28%	28%	28%	28.40%	0.61
KAZE	(632,702, 64)	800	82%	82%	82%	81.85%	0.90	32%	32%	32%	31.86%	0.63
KAZE Extended	(632,702, 128)	800	82%	82%	82%	82.35%	0.90	24%	24%	24%	24%	0.58
A-KAZE	(202,371, 61)	500	75%	75%	75%	75.35%	0.86	54%	54%	54%	54%	0.75
BRISK	(598,471, 64)	770	49%	49%	49%	48.53%	0.74	23%	23%	23%	22.36%	0.56
SIKA	(1,269,426, 128)	1130	67%	67%	67%	67%	0.83	17%	17%	17%	17.5%	0.54

Double-hand mudras

Descriptors	Vector	Clusters	Random forest				
			Prec.	Rec.	F1-Sco.	Accu. (%)	AUC
SIFT	(670,674, 128)	820	91%	91%	91%	91.12%	0.95
SURF	(622,695, 64)	790	92%	92%	92%	91.89%	0.95
ORB	(637,237, 32)	800	72%	72%	72%	72.78%	0.84
KAZE	(632,702, 64)	800	98%	98%	98%	98.37%	0.99
KAZE Extended	(632,702, 128)	800	98%	98%	98%	98.43%	0.99
A-KAZE	(202,371, 61)	500	87%	87%	87%	88.18%	0.92
BRISK	(598,471, 64)	770	69%	69%	69%	69.4%	0.84
SIKA	(1,269,426, 128)	1130	94%	94%	94%	93.5%	0.96

Table 4 Comparison of the work with other related works on Bharatanatyam mudras

Previous works	Dataset	Method used	Classification accuracy
Anami et al. [3]	2400 images of 24 classes of double-hand mudras	Vertical and horizontal cross section of mudra contour, classified using rule-based classifier	95.25%
Anuja et al. [39]	27 classes of single-hand gesture images from Google images and YouTube videos. 18,992 images after data augmentation	Transfer learning of VGG-16 model	94.56%
Kishore et al. [26]	24 classes of both single- and double-hand gesture images. 120 images were used (5 images of each class)	HOG was feature vector and SVM classifier used	90%
Kishore et al. [26]	24 classes of both single- and double-hand gesture images. 120 images were used (5 images of each class)	SIFT, SURF, LBP and Haar features as a feature vector and SVM classifier	80%
Sriparna Saha et al. [50]	28 classes of single-hand gesture images were used.	Fuzzy L membership function as a feature vector	85.1%
S.Mozarkar et al. [36]	68 images of 13 double-hand gesture images were used.	Area, major axis length, minor axis length, centroid and eccentricity were feature vectors and KNN classifier used.	85.29%
Divya et al. [21]	280 images of 28 single-hand gesture images were used.	Edge orientation histogram and shape based skeletal matching used	Not mentioned
Mohanty et al. [35]	1400 images of 10 single-hand gesture images were used.	SIFT, SURF and BRISK as feature descriptors with SVM classifier	The highest accuracy obtained was with SIFT, 91%
Mohanty et al. [35]	840 images of 14 double-hand gesture images were used.	SIFT, SURF and BRISK as feature descriptors with SVM classifier	The highest accuracy obtained was with SURF, 99.8%
Srimani et al. [25]	230 images of 23 double-hand gesture images were used.	Orientation histogram and shape based skeletal matching used	Not mentioned
Proposed method	15,396 images of 29 single-hand gesture images were used	Attempted with 7 feature descriptors and 7 classifiers	The highest accuracy obtained was with KAZE feature descriptor and RF classifier, 92.67%
Proposed method	13,035 images of 21 double-hand gesture images were used	Attempted with 7 feature descriptors and 7 classifiers	The highest accuracy obtained was with KAZE feature descriptor and RF classifier, 98.43%

SURF) for the dataset along with the classifier will have to be evaluated. This can be used for studying the suitability of developing mobile-based applications for identifying the mudras. E-learning applications can also be developed.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human and animal rights The authors had collected hand gesture images from voluntary participants. They were made aware of the fact that the images collected were for research purposes.

References

1. Alcantarilla, Pablo Fernández, Bartoli, Adrien, Davison, Andrew J.: Kaze features. in European Conference on Computer Vision, pp. 214–227, Springer(2012). https://doi.org/10.1007/978-3-642-33783-3_16.
2. Alcantarilla, P.F., Solutions, T.: Fast explicit diffusion for accelerated features in nonlinear scale spaces. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1281–1298 (2011)
3. Anami, B.S., Bhandage, V.A.: A vertical-horizontal-intersections feature based method for identification of Bharatanatyam double hand mudra images. Multimed. Tools Appl. **77**(23), 31021–31040 (2018)
4. Azhar, R., Tuwohingide, D., Kamudi, D., Suciati, N., et al.: Batik image classification using sift feature extraction, bag of features and support vector machine. Proc. Comput. Sci. **72**, 24–30 (2015)
5. Balntas, V., Lenc, K., Vedaldi, A., Mikolajczyk, K.: Hpatches: a benchmark and evaluation of handcrafted and learned local descriptors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5173–5182 (2017)
6. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: speeded up robust features. In: European Conference on Computer Vision, pp. 404–417. Springer, Berlin (2006). https://doi.org/10.1007/11744023_32
7. Bedruz, R.A.R, Fernando, A., Bandala, A., Sybingco, E., Dadios, E.: Vehicle classification using Akaze and feature matching approach and artificial neural network. In: TENCON 2018 (2018 IEEE Region 10 Conference), pp. 1824–1827. IEEE (2018)
8. Belson, W.A.: Matching and prediction on the principle of biological classification. J. R. Stat. Soc. Ser. C (Appl. Stat.) **8**(2), 65–75 (1959)
9. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, Berlin (2006)
10. Boswell, D.: Introduction to support vector machines. Department of Computer Science and Engineering, University of California, San Diego (2002)
11. Boulkenafet, Z., Komulainen, J., Hadid, A.: Face antispoofing using speeded-up robust features and fisher vector encoding. IEEE Signal Process. Lett. **24**(2), 141–145 (2016)
12. Bourlard, H., Kamp, Y.: Auto-association by multilayer perceptrons and singular value decomposition. Biol. Cybern. **59**(4–5), 291–294 (1988)
13. Brunet, D., Vrscay, E.R., Wang, Z.: On the mathematical properties of the structural similarity index. IEEE Trans. Image Process. **21**(4), 1488–1499 (2011)
14. Chien, H.-J., Chuang, C.-C., Chen, C.-Y., Klette, R.: When to use what feature? SIFT, SURF, ORB, or A-KAZE features for monocular visual odometry. In: The Proceedings of the International Conference on Image and Vision Computing New Zealand (IVCNZ), pp. 1–6. IEEE (2016)
15. Coomaraswamy, A., Duggirala, G.K.: The Mirror of Gestures: Being the Abhinayadarpana of Nandikeswara (English Translation). Harvard University Press, London (1917)
16. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
17. Gangrade, J., Bharti, J., Mulye, A.: Recognition of Indian sign language using orb with bag of visual words by Kinect sensor. IETE J. Res. (2020). https://doi.org/10.1080/03772063.2020.1739569
18. Geron, A.: Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. O’Reilly Media, Sebastopol, California (2017)
19. Ghosh, M.: Natyasastra (English Translation). Bibliotheca Indica, Manisha Granthalaya, India (1956)
20. Gupta, S., Thakur, K., Kumar, M.: 2D-human face recognition using SIFT and SURF descriptors of face’s feature regions. Vis. Comput. 1–10 (2020)
21. Hariharan, D., Acharya, T., Mitra S.: Recognizing hand gestures of a dancer. In: The Proceedings of the International Conference on Pattern Recognition and Machine Intelligence, pp.186–192. Springer, Berlin (2011)
22. Huanqing, L., Yi, X., Sizhou, F.: UAV image registration algorithm using color invariant and AKAZE feature. Acta Geodaetica et Cartographica Sinica **46**(7), 900–909 (2017)
23. Hui, J., Yang, Y., Hui, Y., Luo, L.: Research on identify matching of object and location algorithm based on binocular vision. J. Comput. Theor. Nanosci. **13**(3), 2006–2013 (2016)
24. Jisha Raj, R., Dharan, S., Sunil, T.T.: Dimensionality reduction and visualization of Bharatanatyam mudras. Int. J. Image Graph. (2022)
25. Kavitha, S., Srimani, P.K.: Recognizing Samyuktha hand gestures of Bharatanatyam using skeleton matching and gradient orientation. Int. J. Curr. Res. **5**(6), 1457–1462 (2013)
26. Kumar, K.V.V., Kishore, P.V.V.: Indian classical dance mudra classification using HOG features and SVM classifier. Int. J. Electr. Comput. Eng. (2088-8708) **7**(5) (2017)
27. Leutenegger, S., Chli, M., Siegwart, R.Y.: Brisk: binary robust invariant scalable keypoints. In: The Proceedings of the International Conference on Computer Vision, pp. 2548–2555. IEEE (2011)
28. Li, J., Wang, Y., Wang, Y.: Visual tracking and learning using speeded up robust features. Pattern Recogn. Lett. **33**(16), 2094–2101 (2012)
29. Liu, Q., Hui, J., Luo, L., Yang, Y.: Target identification and location algorithm based on surf-brisk operator. Int. J. Pattern Recogn. Artif. Intell. **30**(06), 1655016-1–1655016-11 (2016)
30. Liu, Y., Lan, C., Li, C., Mo, F., Wang, H.: S-AKAZE: an effective point-based method for image matching. Optik **127**(14), 5670–5681 (2016)
31. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
32. Mandal, B., Wang, Z., Li, L., Kassim, A.A.: Performance evaluation of local descriptors and distance measures on benchmarks and first-person-view videos for face identification. Neurocomputing **184**, 107–116 (2016)
33. Mantovani, R.G., Horváth, T., Cerri, Ri., Junior, S.B., Vanschoren, J., de Leon Ferreira de Carvalho, A.C.P.: An empirical study on hyperparameter tuning of decision trees. arXiv preprint arXiv:1812.02207 (2018)
34. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2005)
35. Mohanty, A., Vaishnavi, P., Jana, P., Majumdar, A., Ahmed, A., Goswami, T., Sahay, R.R.: Nrityabodha: towards understanding Indian classical dance using a deep learning approach. Signal Process. Image Commun. **47**, 529–548 (2016)
36. Mozarkar, S., Warnekar, C.S.: Recognizing Bharatnatyam mudra using principles of gesture recognition gesture recognition. Int. J. Comput. Sci. Netw. **2**(2), 46–52 (2013)
37. Okawa, M.: Synergy of foreground–background images for feature extraction: offline signature verification using fisher vector with fused KAZE features. Pattern Recogn. **79**, 480–489 (2018)
38. Ordóñez, Á., Argüello, F., Heras, D.B.: Alignment of hyperspectral images using KAZE features. Remote Sens. **10**(5), 756 (2018). https://doi.org/10.3390/rs10050756
39. Parameshwaran, A.P., Desai, H.P., Sunderraman, R., Weeks, M.: Transfer learning for classifying single hand gestures on comprehensive Bharatanatyam Mudra dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (2019)
40. Parashar, D., Agrawal, D.: Automated classification of glaucoma stages using flexible analytic wavelet transform from retinal fundus images. IEEE Sens. J. **20**(21), 12885–12894 (2020)

41. Parashar, D., Agrawal, D.: 2-D compact variational mode decomposition-based automatic classification of glaucoma stages from fundus images. IEEE Trans. Instrum. Meas. **70**, 1–10 (2021)
42. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
43. Pieropan, A., Björkman, M., Bergström, N., Kragic D.: Feature descriptors for tracking by detection: a benchmark. arXiv preprint arXiv:1607.06178 (2016)
44. Pinto, B., Anurenjan, P.R.: Video stabilization using speeded up robust features. In: The Proceedings of the International Conference on Communications and Signal Processing, pp. 527–531. IEEE (2011)
45. Probst, P., Wright, M.N., Boulesteix, A.-L.: Hyperparameters and tuning strategies for random forest. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **9**(3), e1301 (2020)
46. Raileanu, L.E., Stoffel, K.: Theoretical comparison between the Gini index and information gain criteria. Ann. Math. Artif. Intell. **41**(1), 77–93 (2004)
47. Ramachandrasekhar, P.: Abhinayadarpanam, pp. 71–126. Giri Trading Agency Private Limited, India (2013)
48. Rish, I., et al.: An empirical study of the Naive Bayes classifier. In: The Proceedings of the IJCAI Workshop on Empirical Methods in Artificial Intelligence, vol. 3, pp. 41–46 (2001)
49. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: "Orb: an efficient alternative to sift or surf". In: The Proceedings of the International Conference on Computer Vision, pp. 2564–2571. IEEE (2011)
50. Saha, S., Ghosh, L., Konar, A., Janarthanan, R.: Fuzzy l membership function based hand gesture recognition for Bharatanatyam dance. In: The Proceedings of the 5th International Conference on Computational Intelligence and Communication Networks (CICN), pp. 331–335. IEEE (2013)
51. Salahat, E., Qasaimeh, M.: Recent advances in features extraction and description algorithms: a comprehensive survey. In: The Proceedings of the IEEE International Conference on Industrial Technology (ICIT), pp. 1059–1063 (2017)
52. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition. In: The Proceedings of the 15th ACM International Conference on Multimedia, pp. 357–360 (2007)
53. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep fisher networks for large-scale image classification. In: The Proceedings of the Advances in Neural Information Processing Systems, pp. 163–171 (2013)
54. Srivastava, S., Mukherjee, P., Lall, B.: Characterizing objects with sika features for multiclass classification. Appl. Soft Comput. **46**, 1056–1066 (2016)
55. Verma, N.K., Goyal, A., Vardhan, A.H., Sevakula, R.K., Salour, A.: Object matching using speeded up robust features. In: The Proceedings of the Intelligent and Evolutionary Systems, pp. 415–427. Springer, Berlin (2016)
56. Wang, Z., Lu, B., Chi, Z., Feng, D.: Leaf image classification with shape context and sift descriptors. In: The Proceedings of the IEEE International Conference on Digital Image Computing: Techniques and Applications, pp. 650–654 (2011)
57. Wikipedia contributors, Bharatanatyam—Wikipedia, the free encyclopedia, 2020. [Online; accessed 1-December-2020]. Available: https://en.wikipedia.org/w/index.php?title=Bharatanatyam&oldid=992081903
58. Yang, Y., Newsam, S.: Comparing SIFT descriptors and Gabor texture features for classification of remote sensed imagery. In: The Proceedings of the 15th IEEE International Conference on Image Processing, pp. 1852–1855 (2008)
59. Zhong, B., Li, Y.: Image feature point matching based on improved sift algorithm. In: The Proceedings of the IEEE 4th International Conference on Image, Vision and Computing (ICIVC), pp. 489–493 (2019)

R. Jisha Raj received her B.Tech. in electronics and communication engineering from Government Engineering College, Barton Hill, Thiruvananthapuram (Kerala University), India, in 2003. She obtained her ME degree in 2005 from Anna University, India. Currently, she is pursuing her Ph.D. degree from the Electronics Engineering Department, College of Engineering, Chengannur (Affiliated to APJ Abdul Kalam Technological University), Kerala, India. Her research interests are computer vision, machine learning and image processing.

Smitha Dharan obtained B.Tech in Computer Science and Engineering from University of Kerala in 1992; MTech in Computer and Information Science in 2001 from Cochin University of Science and Technology; and PhD in 2010 from University of Kerala. Currently, she is with Dept. of Computer Science and Engineering, College of Engineering Chengannur, Kerala, India, under the Institute of Human Resources Development (IHRD). Her areas of interest are bioinformatics, image processing and data mining.

T. T. Sunil received his B.Tech. degree in electronics and communication engineering in 1990 from Mar Athanatius College, Kothamangalam, Kerala, India, and M.Tech. degree in computer and information sciences from Cochin University of Science and Technology, Kerala, in 2000. He received his Ph.D. from the Department of Electrical Engineering, IIT Bombay, India, in 2015. Currently, he is with Electronics Engineering Department, College of Engineering, Attingal, Thiruvananthapuram, Kerala, India, under the Institute of Human Resources Development (IHRD). His research interests are computer vision, machine learning and signal processing.