Pawan Kumar Singh*, Supratim Das, Ram Sarkar and Mita Nasipuri

Feature Selection Using Harmony Search for Script Identification from Handwritten Document Images

DOI 10.1515/jisys-2016-0070
Received June 1, 2016; previously published online March 16, 2017.

Abstract: The feature selection process can be considered a problem of global combinatorial optimization in machine learning, which reduces the irrelevant, noisy, and non-contributing features, resulting in acceptable classification accuracy. Harmony search algorithm (HSA) is an evolutionary algorithm that is applied to various optimization problems such as scheduling, text summarization, water distribution networks, vehicle routing, etc. This paper presents a hybrid approach based on support vector machine and HSA for wrapper feature subset selection. This approach is used to select an optimized set of features from an initial set of features obtained by applying Modified log-Gabor filters on prepartitioned rectangular blocks of handwritten document images written in either of 12 official *Indic* scripts. The assessment justifies the need of feature selection for handwritten script identification where local and global features are computed without knowing the exact importance of features. The proposed approach is also compared with four well-known evolutionary algorithms, *namely* genetic algorithm, particle swarm optimization, tabu search, ant colony optimization, and two statistical feature dimensionality reduction techniques, *namely* greedy attribute search and principal component analysis. The acquired results show that the optimal set of features selected using HSA gives better accuracy in handwritten script recognition.

Keywords: Harmony search algorithm, wrapper feature selection, handwritten script identification, *Indic* scripts, Modified log-Gabor filters, support vector machine.

1 Introduction

Many strategies have been exploited for the task of feature selection (FS), in an effort to identify more compact and better-quality feature subsets. The development of nature-inspired stochastic search techniques allows multiple good-quality feature subsets to be discovered without resorting to exhaustive search. Selecting a subset of features from the entire feature set is often beneficial because it lessens the computation time along with helping achieve better classification accuracy. At a higher dimension, it becomes very difficult to visualize the data and there may be some non-contributing features that may even mislead the classifier's decision [31]. Identification of those features at higher dimension, though a challenging task, motivates the researchers to work in this domain. For a large feature dimensionality, it is almost impossible to check the accuracies considering all possible feature subsets. Thus, one may apply the hill climbing (HC) method by successively adding/removing feature(s) into/from the original feature set. However, the HC method gets stuck at local minima. Thus, the need to explore the search space using some heuristic search techniques will be of utmost importance. Practical problems that arise during analyzing the real-world data are often related to the number of features (so called *curse of dimensionality*). The inability of the classification tools to identify and

***Corresponding author: Pawan Kumar Singh,** Department of Computer Science and Engineering, Jadavpur University, 188, Raja S.C. Mullick Road, Kolkata 700032, West Bengal, India, e-mail: pawansingh.ju@gmail.com
Supratim Das, Ram Sarkar and Mita Nasipuri: Department of Computer Science and Engineering, Jadavpur University, 188, Raja S.C. Mullick Road, Kolkata 700032, West Bengal, India

extract patterns or rules easily may also be attributed to high interdependency among individual features, or the behavior of the combined features. Techniques such as text processing and classification can benefit greatly from FS, once the noisy, irrelevant, redundant, or misleading features are removed. Given a feature set with size n, there will be 2^n possible feature subsets. The FS problem is to find a minimal feature subset while retaining a suitably high accuracy in classifying the data [25].

The FS approaches can generally be divided into two groups: filter and wrapper approaches. The filter approach operates independently of any learning algorithm. These methods rank the features by some criteria and neglect all features that do not achieve a sufficient score. Due to their computational efficiency, the filter methods are very popular for high-dimension data. Some popular filter methods are *F*-score criterion [7], mutual information [40], information gain [1], correlation [17], etc. On the other hand, the wrapper approaches involve a predetermined learning model while selecting features on measuring the parameters of the particular machine learning model [17]. Filter approaches are usually less computationally intensive than wrappers, but they produce a feature set that is biased to a specific type of predictive model. This unbiased tuning means a feature set from a filter is more general than the set from a wrapper, usually giving lower prediction performance than a wrapper. However, the feature set selected using the former method does not contain the assumptions of a prediction model, and so is more useful for exposing the relationships among the original features.

The rest of the paper is organized as follows: Section 2 presents a brief review of some of the previous approaches to FS, and Section 3 illustrates the motivation behind the proposed work. Section 4 describes the Modified log-Gabor filter-based feature extraction procedure for handwritten *Indic* script identification, whereas in Section 5, the basics of the harmony search (HS)-based feature subset selection process are discussed. Section 6 describes the performance of the proposed script identification system, and finally, Section 7 concludes the work.

2 Related Works

There has been an immense amount of work in the area of FS, which credits its importance. Sivagaminathan and Ramakrishnan [35] presented a hybrid method based on ant colony optimization (ACO) and artificial neural networks to address FS. The proposed hybrid model was demonstrated using datasets from the domain of medical diagnosis, yielding promising results. El Alami proposed a novel text categorization algorithm [9] called chaos genetic FS optimization in order to choose a subset of available features by eliminating unnecessary features to the classification task. The proposed algorithm selected the optimal subsets in both empirical and theoretical works in machine learning, and presented a general framework for text categorization. Experimental results showed that the proposed algorithm simplified the FS process effectively and obtained higher classification accuracy with only a subset of original features. Bermejo et al. [3] proposed a stochastic algorithm based on the greedy randomized adaptive search procedure (GRASP) meta-heuristic, with the main goal of speeding up the feature subset selection process, basically by reducing the number of wrapper evaluations to be carried out. GRASP is a multi-start constructive method that formulates a solution in its first stage, and then applies a modifier stage on that solution for an improved result. Several instances of the proposed GRASP method were experimentally tested and compared with state-of-the-art algorithms over 12 high-dimensional datasets. The statistical analysis of the results illustrated that the proposal was comparable in accuracy and cardinality of the selected subset to previous algorithms, but required significantly fewer evaluations. Forsati et al. [11] formulated the problem of FS as an optimization problem and introduced a novel FS procedure in order to achieve better classification results. Experiments over a standard benchmark demonstrated that applying bee colony optimization in the context of FS was a feasible approach. Das et al. [4] proposed a methodology where local regions of varying heights and widths were created dynamically on the pattern images. Genetic algorithm (GA) was then applied on these local regions to sample the optimal set of local regions containing best distinguishing information among the pattern classes. An optimal feature set was then extracted from those selected regions. Other popular optimization techniques like simulated

annealing and HC had also been evaluated and, finally, a comparison with the proposed technique was done on a dataset of handwritten *Bangla* digits. Vieira et al. [39] developed modified binary particle swarm optimization (MBPSO) for feature subset selection. It optimized the kernel parameters of support vector machine (SVM) as well as controlled the premature convergence of the algorithm. Zeng et al. [41] introduced a novel hybrid FS method based on rough conditional mutual information and naive Bayesian classifier, which could also be used to filter the irrelevant and redundant features. Subsequently, to reduce the feature and improve classification accuracy, a wrapper approach based on naive Bayesian classifier was used to search the optimal feature subset in the space of a candidate feature subset that was selected by filter model. The results showed that the approach obtained not only high classification accuracy, but also selected the least number of features. Nazir et al. [27] proposed an efficient gender classification technique for real-world face images (labeled faces in the wild). This work extracted facial local features using local binary pattern (LBP), and then these features were fused with clothing features to enhance the classification accuracy. In the following step, PSO and GA were combined to select the most important features set that more clearly represented the gender, and thus the data dimensionality was reduced. Roy et al. [29] implemented artificial bee colony optimization to pin point the set of local regions offering the optimal discriminating feature set for handwritten numeral and character recognition. Initially, eight-directional gradient features were extracted from every region of different levels of partitions created using a center of gravity-based quad-tree partitioning approach. Then, using this approach, at each level, the sampling process was done based on SVM in every single region. De Stefano et al. [5] presented a novel GA-based FS algorithm in which feature subsets were evaluated by means of a specifically devised separability index. This index measured the statistical properties of the feature subset and did not depend on any specific classification scheme. The proposed index represented an extension of the Fisher linear discriminant method and used covariance matrices for estimating how class probability distributions were spread out in the considered N-dimensional feature space. A key property of the approach was that it does not require any *a priori* knowledge about the number of features to be used in the feature subset. Roy et al. [30] proposed a new FS methodology on the basis of features combined class separability power, using the framework of axiomatic fuzzy set theory, which provided the rules for logic operations needed to interpret the combinations of features from the fuzzy feature set. Based on these combinational rules, the class separability power of the combined features was determined, and subsequently the most powerful subset of the feature set was selected. Mohammadi and Abadeh [26] introduced a new feature-based blind steganalysis method for detecting stego images from the cover images in JPEG images using an FS technique based on the artificial bee colony optimization technique. Kashef and Nezamabadi-pour [18] presented a novel feature selection algorithm based on ACO, called advanced binary ACO. Features were treated as graph nodes to construct a graph model and were fully connected to each other. The performance of the proposed algorithm was compared with the performance of binary GA, binary PSO (BPSO), catfish BPSO, improved binary gravitational search algorithm, and some prominent ACO-based algorithms on the task of FS on 12 well-known UCI datasets. Simulation results verified that the algorithm provided a suitable feature subset with good classification accuracy using a smaller feature set than competing FS methods. Ahmad [2] used principal component analysis (PCA) for feature reduction and subset selection in which features were primarily projected into a principal space and then elected based on their eigenvalues, but the feature with the highest eigenvalue may not had the guarantee to provide optimal sensitivity for the classifier. To avoid this problem, two evolutionary optimization approaches like GA and PSO had been used to search the most discriminative subset of transformed features. Finally, it was observed that a feature subset selection based on PSO provided better performance as compared to GA. Singh et al. [32] proposed a PSO- and F-score-based FS algorithm for selecting the significant features that contributed to improve the classification accuracy. The performance of their proposed method was evaluated with various classifiers such as SVM, naive Bayes, k-NN, and decision tree. The experimental results illustrated that the proposed method outperformed some existing methods.

In particular, HS is a recently developed technique mimicking musicians' experiences, which has been effectively utilized to cope with FS problems. Kumar et al. [22] presented a novel variance-based HS (VHS) algorithm for solving optimization problems. The problem of setting a constant parameter in the algorithm

was removed, and the technique was named as explorative HS. The proposed algorithm was then employed for a data clustering problem on four real-life datasets chosen from the UCI machine learning repository. The results indicated that the VHS-based clustering outperformed the existing well-known clustering algorithms. Kumar et al. [21] introduced a parameter adaptive HS (PAHS) algorithm for solving optimization problems. The two important parameters of HS algorithm (HSA), namely harmony memory consideration rate (HMCR) and pitch adjusting rate (PAR), were allowed to dynamically change in their proposed algorithm in order to get the global optimal solution. Evaluation of the proposed algorithm for data clustering on five real-life UCI datasets showed better results than other algorithms. Kumar et al. [23] assumed the problem of automatic data clustering as the searching of optimal number of clusters in order to optimize the attained partitions. The PAHS algorithm (as described in Ref. [21]) was employed as an underlying optimization strategy. The data points of the different cluster centroids were assigned on the basis of newly developed weighted Euclidean distance. The algorithm was compared with four existing clustering techniques, and the same was further applied for automatic segmentation of grayscale and color images. Kumar et al. [20] proposed a novel automatic unsupervised FS method based on gravitational search algorithm, called AFSGSA (automatic FS using the gravitational search algorithm). The algorithm requires no prior knowledge of the data to be classified and the number of features to be selected. A novel fitness function was also included in order to make the search more efficient. Comparison of the performance of AFSGSA concluded the efficiency and efficacy of the proposed FS technique over other existing techniques for the same. In this paper, we present a wrapper FS method based on HS to discover a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original data in the feature space. Applying this FS method, we have achieved some significant improvement in the recognition accuracy of the Modified log-Gabor filter-based handwritten *Indic* script identification technique.

3 Motivation

India is a multilingual country with 23 constitutionally recognized languages [38] written in 12 major scripts. Besides these, hundreds of other languages are used in India, each one with a number of dialects. The officially recognized languages are *Hindi*, *Bengali*, *Punjabi*, *Marathi*, *Gujarati*, *Oriya*, *Sindhi*, *Assamese*, *Nepali*, *Urdu*, *Sanskrit*, *Tamil*, *Telugu*, *Kannada*, *Malayalam*, *Kashmiri*, *Manipuri*, *Konkani*, *Maithali*, *Santhali*, *Bodo*, *Dogari*, and *English*. The 12 major scripts used to write these languages are *Devanagari*, *Bangla*, *Oriya*, *Gujarati*, *Gurumukhi*, *Tamil*, *Telugu*, *Kannada*, *Malayalam*, *Manipuri*, *Roman*, and *Urdu*. Of these, *Urdu* is derived from the *Persian* script and is written from right to left. The first 10 scripts are originated from the early *Brahmi* script (300 BC) and are also referred to as *Indic* scripts. Due to the fact that any optical character recognition (OCR) system can recognize only a script of particular type, it is impossible to design a single recognizer that can identify a variety of scripts/languages. An alternate solution to handle this problem is to use a pool of OCRs corresponding to different scripts. This, in turn, requires the identification of the document's script before feeding it to the corresponding OCR system. Here lies the problem of script identification. In general, script identification can be achieved at any of the three levels: (a) page level, (b) text-line level, and (c) word level. Identifying scripts at page level can sometimes be too convoluted and protracted due to large computational complexity. Again, the identification of scripts at text-line or word level requires the accurate segmentation of the document pages into their corresponding text-lines and words. That is, the accuracy of the script identification, in turn, depends on the accuracies of its text-line and word segmentation methods, respectively. In addition, identifying text words of different scripts with only a few numbers of characters may not always be feasible because at word level, the number of characters present in a single word may not be always informative. Thus, in the present work, we have performed script identification at block level, where entire document pages are decomposed gradually into their corresponding smaller subblocks using a quad-tree-based page subdivision approach.

A comprehensive literature survey on *Indic* script identification (described in Ref. [34]) indicates that a lot of researchers have contributed to the handwritten script identification problem by inventing/deriving

a bundle of features; however, the exploration of the optimum feature set for this problem has not received any attention from the research community till date in spite of its importance. In general, it is noticed that heuristic search techniques, *namely* GA, PSO, ACO, and TS, have mostly been applied to reduce the complexity of the FS process. However, in this paper, an HS-based FS procedure is described instead, owing to its simplicity and stochastic behavior (which helps in escaping from trapping at local optima). In comparison with GA-, ACO-, TS-, and PSO-based searches, HS is less explored by the research community, although it has gained huge popularity in recent times. The present HS-based search is also compared with some preceding entrenched optimization methods, and the experimental results also confirm the effectiveness of HS-based search.

4 Proposed Work

The proposed work employs a two-pass approach. In the first pass, the texture-based features are extracted by applying Modified log-Gabor filters [33] designed at multiple scales and multiple orientations. These features are estimated at different sub-image levels of the document images decomposed by the quad-tree-based approach. This approach is recursively applied to subdivided image segments until the image segment does not provide any significant information. Thus, at the first level, the whole document image is divided into four equal blocks (L_1, L_2, L_3, and L_4), whereas at the second level, L_1 block is again subdivided into another four subblocks, *namely* L_{11}, L_{12}, L_{13}, and L_{14}. Similarly, each of the other blocks, i.e. L_2, L_3, and L_4, are also subdivided into four subblocks. Thus, at second-level decomposition, a total of 16 blocks is realized. Therefore, the total number of subblocks at l-th level can be written as 4^l, where l denotes the level of decomposition. For an image of size $M \times N$, the size of each subblock at l-th level is $(M/2^l) \times (N/2^l)$. For the present work, the maximal value of l is chosen to be 4. Figure 1 shows the quad-tree decomposition for a handwritten *Tamil* document image. It is evident from Figure 1E that the text block at fourth level has just half the word images of two consecutive text-lines. If the page decomposition is performed beyond fourth level, we would be left with hardly one or two characters of the word image, which may not be sufficient for the script identification purpose. Thus, we limit our quad-tree-based approach for page decomposition till $l = 4$.

In the second pass, the HSA is applied to select the feature subset where the objective function is the performance of the SVM classifier. The block diagram of the proposed two-pass approach is shown in Figure 2.

4.1 Modified Log-Gabor Filters

The log-Gabor function was proposed by Field [10] as a means to overcome the limitations of Gabor function. Field suggested that when viewed on a logarithmic frequency scale, coding of natural images by filters having Gaussian transfer functions is superior. The log-Gabor function has a transfer function of the following form:

$$G(w) = e^{(-\log(w/w_0)^2)} \Big/ (2\,(\log(k/w_0)^2) \tag{1}$$

where w_o is the filter's center frequency. To obtain constant shape ratio filters, the term k/w_o must also be held constant for all chosen values of w_o.

The log-Gabor functions do not have any DC component, and their transfer function has an extended tail at the high frequency end. The studies of Field on the statistics of ordinary images claim that these images have amplitude spectra that gradually fall off at approximately $1/w$. In order to encode those images that possess such spectral characteristic filters, similar spectra must be used. Further, log-Gabor functions having extended tails are able to encode images more efficiently than ordinary Gabor functions, as these log-Gabor filters neither overrepresent the lower-frequency components nor underrepresent the high-frequency components in any encoding.

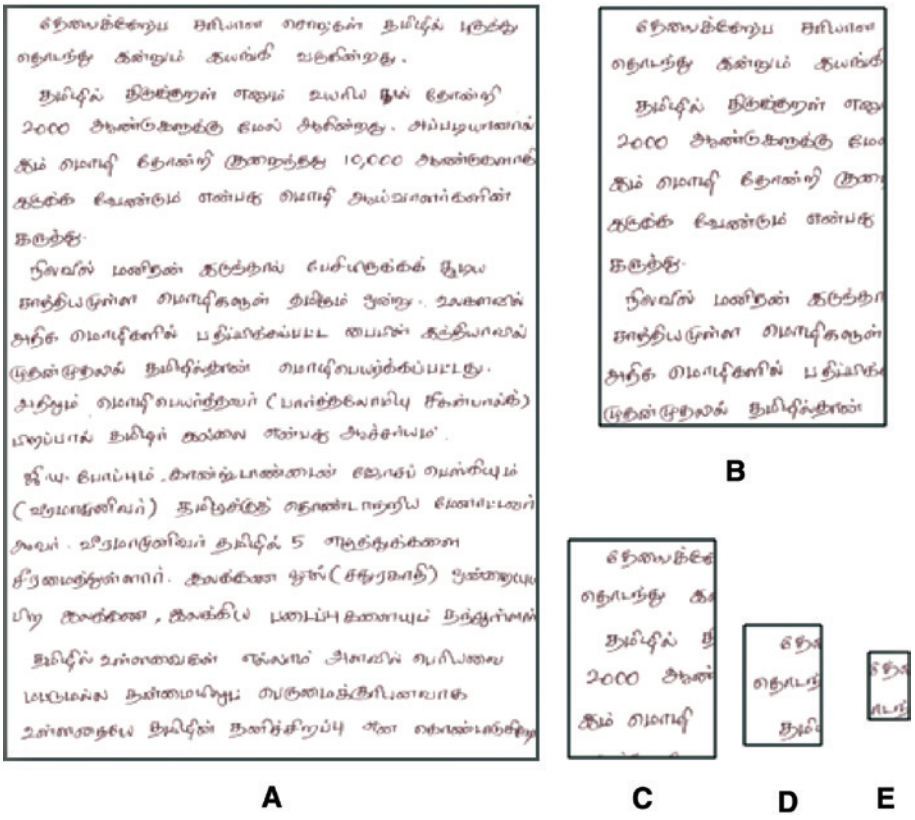

Figure 1: Illustration of (A) Handwritten *Tamil* Document Page after (B) First-Level (L_1), (C) Second-Level (L_{11}), (D) Third-Level (L_{111}), and (E) Fourth-Level (L_{1111}) Decomposition.

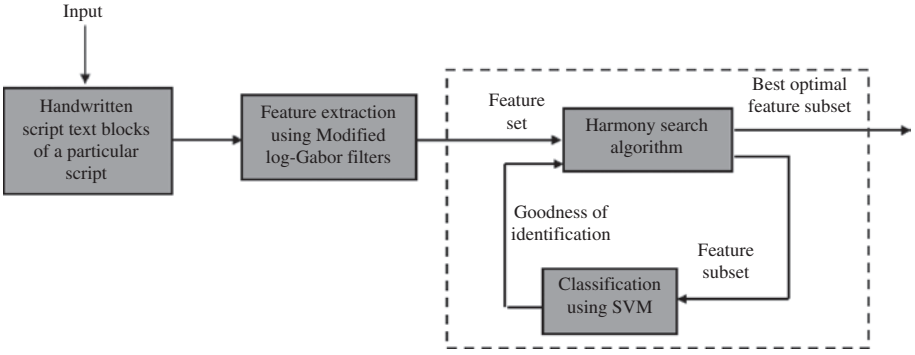

Figure 2: Schematic Diagram of the Proposed Two-Pass Approach for Handwritten Script Identification.

For proper texture analysis of an image using the features extracted using log-Gabor filter, the frequency components are necessary. In order to retain the spatial information eliminated by a simple Fourier transform, a windowed Fourier transform (WFT) is used. WFT involves multiplication of the image $X(n_0, n_1)$ by the window function $w(n_0, n_1)$ followed by application of the Fourier transform on the result to obtain

$$\begin{aligned}\tau(n_0^{(0)}, n_1^{(0)}, u_0, u_1) &= \int_{-\infty}^{\infty} X(n_0, n_1) w(n_0 - n_0^{(0)}, n_1 - n_1^{(0)}) e^{-2\pi i(u_0 n_0 + u_1 n_1)} dn_0 dn_1 \\ &= e^{-2\pi i(u_0 n_0^{(0)} + u_1 n_1^{(0)})} [X(n_0^{(0)}, n_1^{(0)}) * (w(n_0^{(0)}, n_1^{(0)}) e^{-2\pi i(u_0 n_0^{(0)} + u_1 n_1^{(0)})})] \\ &= e^{-2\pi i(u_0 n_0^{(0)} + u_1 n_1^{(0)})} [X * m_{u_0, u_1}](n_0^{(0)}, n_1^{(0)}). \end{aligned} \quad (2)$$

Here, $(n_0^{(0)}, n_1^{(0)})$ is the position in the image where the horizontal and vertical frequencies u_0 and u_1 are computed. The WFT is basically a convolution of the image with the filter m_{u_0,u_1}. For texture analysis, both spatial and frequency information are desired. Thus, we try to achieve a good trade-off between these two with respect to the uncertainty principle of Fourier transform. Gabor transforms use a Gaussian function as the optimally concentrated function in the spatial as well as in the frequency domain [10]. Here, non-isotropic Gaussian is of the form

$$m_{f,\varnothing}(n_0, n_1) = \frac{1}{2\pi\sigma^2\lambda} e^{-\frac{1}{2\sigma^2}\left(\frac{n_0^{2'}}{\lambda^2}+n_1^{2}\right)} e^{2\pi i f n_0'} \tag{3}$$

$$M_{f,\varphi}(u_0, u_1) = e^{-2\pi^2\sigma^2[(u_0'-f)^2\lambda^2+u_1'^2]} \tag{4}$$

with the center frequency $f = \sqrt{u_0^2 + u_1^2}$ and the rotated coordinates $(\acute{n}_0, \acute{n}_1) = (n_0 \cos \varnothing + n_1 \sin \varnothing - n_0 \sin \varnothing + n_1 \cos \varnothing))$.

Here, $1/\lambda$ is the aspect ratio.

Due to the convolution theorem, the filter interpretation of the Gabor transform allows the efficient computation of the Gabor coefficients $G_{f,\varnothing}(n_0, n_1)$ by multiplication of the Fourier transformed image $\tau(u_0, u_1)$ with the Fourier transform of the Gabor filter $M_{f,\varnothing}(u_0, u_1)$. The inverse Fourier transform is then applied on the resultant vector as defined below:

$$G_{f,\varphi}(n_0, n_1) = FFT^{-1}\{\tau(u_0, u_1).M_{f,\varnothing}(u_0, u_1)\} \tag{5}$$

The images, after low pass filtering, are passed as input to a function that computes the Gabor energy feature from them. This is done by multiplying in frequency domain [which is equivalent to two-dimensional (2D) convolution in the time-space domain]. The input image is then passed to a function to yield a Gabor array, which is the array equivalent of the image after Gabor filtering. The function displays the image equivalent of the magnitude and the real part of the Gabor array pixels.

Consider that there are n_s scales and n_o number of orientations, resulting in $n_s \times n_o$ different filters. Let J denote the Fourier transform of the input image, $G_{s,o}$ the Gabor filter at scale s and orientation o, and $V_{s,o}$ the output of the convolution of $G_{s,o}$ and J.

$$V_{s,o} = J * G_{s,o}. \tag{6}$$

The local responses of each of the Gabor filters can also be represented in terms of amplitude $A_{s,o}(x, y)$ and energy $E_{s,o}(x, y)$, as defined below:

$$A_{s,o} = |V_{s,o}(x, y)| \tag{7}$$

and

$$E_{s,o}(x, y) = |Real\{V_{s,o}(x, y)\}| - |Img\{V_{s,o}(x, y)\}|, \tag{8}$$

where (x, y) denotes the 2D coordinates of a pixel, and *Real* and *Img* denote the real and imaginary parts of the filter responses, respectively. Next, the median overall orientations for a fixed scale s for $A_{s,o}$ and $E_{s,o}$ are calculated as follows [14]:

$$A_s(x, y) = median\{o = 1, 2\ldots, n_o\}A_{s,o}(x, y) \tag{9}$$

and

$$E_s(x, y) = median\{o = 1, 2\ldots, n_o\}E_{s,o}(x, y). \tag{10}$$

For the present work, features based on Modified log-Gabor filters have been extracted for five scales (n_s=1, 2, 3, 4, and 5) and 12 orientations (n_o=0°, 15°, 30°, … …., 165°, 180°), where each filter has to be convolved with the input image to obtain 60 (i.e. 12 * 5) different representations (response matrices) for the same image.

These response matrices are then converted to feature vectors. Each input image provides us with one feature vector consisting each of 60 elements calculated using amplitude $A_s(x, y)$ and energy $E_s(x, y)$ values. Finally, a set of 120 multiscale and multidirectional oriented features has been used for the block-level recognition of 12 official scripts.

The algorithm of the feature extraction methodology can be described as follows:

Step 1: Input an RGB document image and convert it into a grayscale image $I(x, y)$, where x and y are the spatial coordinates of the image.

Step 2: Select an integer value l, where l indicates the level of decomposition. For the present work, the optimal value of l is varied from 2 to 4.

Step 3: Perform l-level quad-tree decomposition to generate 4^l blocks. Feature values are generated from each of these segmented blocks.

Step 4: Convolve the block image of step 3 with modified log-Gabor filter with five different scales ($n_s = 1, 2, 3, 4,$ and 5) and 12 different orientations ($n_o = 0°, 15°, 30°, \ldots \ldots, 165°, 180°$). Thus, for each block, we get a total of 60 convolved filter outputs.

Step 5: Acquire the absolute value of these filtered output block images and calculate the amplitude and energy values.

Step 6: Repeat step 3 to step 5 to extract a total of 120 features from each of the script block images.

5 HS Optimization

HS is a music-based meta-heuristic optimization algorithm, proposed by Geem [12] in 2000. It was inspired by the observation that the aspiration of a musician is to search for a perfect state of harmony. The search process in optimization can be compared to a jazz musician's improvisation process. Likewise, an optimal solution to an optimization problem should be the desirable solution available to the problem under the given objectives and limited by constraints. Both processes intend to produce the best or optimum output. In order to explain the HS in more detail, let us first understand the improvisation process by a skilled musician. When a musician is improvising, he or she has three possible choices: (a) play any famous piece of music (a series of pitches in harmony) exactly from his or her memory, (b) play something similar to a known piece (thus adjusting the pitch slightly), or (c) compose new or random notes. Geem et al. [13] formalized these three options into a quantitative optimization process in 2001, and the three corresponding components became usage of harmony memory (HM), pitch adjusting, and randomization.

The usage of HM ensures that the best harmonies pass over to the new HM. This usage is controlled by assigning a parameter δ, known as HM accepting or considering rate. The value of δ normally lies between 0 and 1, but typically, it is set to 0.7. The second component called pitch adjustment, $\rho \in [0.1, 0.5]$, is used to control the degree of the adjustment, and for most applications, this is set to 0.3. The third component called randomization is used to increase the diversity of the solutions. Although adjusting pitch has a similar role with randomization, it is limited to certain local pitch adjustment and thus corresponds to a local search. The use of randomization can drive the system further to explore diverse solutions so as to find the global optimality. It is given by

$$P_{rand} = 1 - \delta \tag{11}$$

Pitch adjustment is determined by a pitch bandwidth P_{band} and a pitch adjusting rate ρ. The linear adjustment of pitch is done in the following way:

$$x_{new} = x_{old} + P_{band} * \varepsilon \tag{12}$$

where x_{old} is the existing pitch or solution from the HM and x_{new} is the new pitch after the pitch adjusting action. This essentially produces a new solution around the existing quality solution by varying the pitch slightly by a small random amount. Here, ε is a random number generator in the range of $[-1, 1]$.

5.1 Characteristics of HSA

While reviewing other meta-heuristic algorithms, it has been found that the success of any optimization process depends on two major contradicting components: diversification and intensification. Proper diversification or exploration makes sure the search in the parameter space can explore as many locations and regions as possible in an efficient and effective manner. It also ensures that the evolving system will not be trapped in biased local optima. On the other hand, the appropriate intensification or exploitation intends to exploit the history and experience of the search process. Its aim is to ensure to speed up the convergence by reducing the randomness and limiting diversification.

The two subcomponents, namely pitch adjustment and randomization, control the diversification in the HSA. These two parameters may contribute toward the high efficiency of the HS method. The level of efficiency of the first subcomponent in composing "new music", or generating new solutions, via randomization would be at least similar in comparison to other algorithms attained by randomization. On the other hand, an additional subcomponent, pitch adjustment denoted by ρ, is calculated by regulating the pitch in the given bandwidth by a small random amount relative to the existing pitch or solution from the HM. Local solutions are refined by using this parameter. The preservation of good local solutions are ensured by both δ and ρ, while the global search space will be efficiently explored by randomization and δ. The vulnerability of this is that it is a controlled diversification around the good solutions (good harmonics and pitches), as well as it operates almost like an escalation factor. The intensification is mainly represented in the HSA by δ. A high δ implies a high chance of the selection or inheritance of good solutions from the memory, whereas a low δ means that the solutions will converge more slowly.

5.2 HSA-Based Feature Subset Selection

The HSA consists of the following five steps [24]:

Step 1: Initialization of the Optimization Problem and Algorithm Parameters
The optimization problem can be defined as follows:

Maximize $f(x)$ subject to $x_i \in X$, $X = x_1, x_2, \ldots \ldots \ldots x_n$, where $f(x)$ is the objective function, X_i is the set of possible range of values for each decision variable, that is $L_{x_i} < X_i < U_{x_i}$, where U_{x_i} and L_{x_i} are the upper and lower bounds of each decision variable. The control parameters of HS are the HM size (HMS), i.e. the number of solution vectors in the HM (in each iteration), and the maximum number of iterations (*Maxltr*) or stopping criterion.

Step 2: HM Initialization
HM is a memory location where all the solution vectors (sets of decision variables) are stored. The HM matrix is initially filled with as many randomly generated solution vectors as the HMS.

$$HM = \begin{bmatrix} x_1^1 & x_2^1 & x_N^1 \\ x_1^2 & x_2^2 & x_N^2 \\ x_1^{HMS} & x_2^{HMS} & x_N^{HMS} \end{bmatrix} \tag{13}$$

Step 3: New Harmony Improvisation
In this step, a new harmony vector $x' = (x_1', x_2', x_3', x_3', \ldots \ldots \ldots, x_N')$ is generated based on three rules: (a) memory consideration, (b) pitch adjustment, and (c) random selection. Generating a new harmony is called "improvisation". In the memory consideration, the value of the first decision variable x_1' for the new vector is chosen from the values already existing in the current HM, i.e. from the set $\{x_1^1, \ldots \ldots, x_1^{HMS}\}$, with a probability δ. Values of the other decision variables $x_2', x_3', x_3', \ldots \ldots \ldots, x_N'$ are also chosen in the same

manner. The value of δ, varying in the range 0 and 1, is the rate of choosing one value from the previous values stored in the HM, while $(1-\delta)$ is the rate of randomly selecting a new value from the possible range of values.

$$x_i' = \begin{cases} x_i \epsilon \{x_i^1, x_i^2, x_i^3, \ldots \ldots \ldots, x_i^{HMS}\} & \text{with probability } \delta \\ x_i \epsilon X_i & \text{with probability}(1-\delta) \end{cases} \tag{14}$$

Every component obtained by the memory consideration is further examined to determine whether it should be pitch adjusted or not. This operation uses the parameter ρ as follows:

$$x_i' = \begin{cases} (x_i' \pm P_{band} * \varepsilon) & \text{with probability } \rho \\ x_i' & \text{with probability}(1-\rho) \end{cases} \tag{15}$$

where P_{band} is the pitch bandwidth and ε is a random number generator in the range of [– 1, 1].

Step 4: Update HM

If the new harmony vector, $x' = (x_1', x_2', x_3', x_3', \ldots\ldots\ldots, x_N')$, is better than the worst harmony in the HM, from the point of view of objective function value, the new harmony is included in the HM and the existing worst harmony is excluded from the same.

Step 5: Repetition of Steps 3 and 4 Until the Stopping Criterion Is Satisfied

If the stopping criterion (i.e. *maxItr*) is satisfied, the computation is terminated. Otherwise, Steps 3 and 4 are repeated.

This is summarized in the flowchart shown in Figure 3.

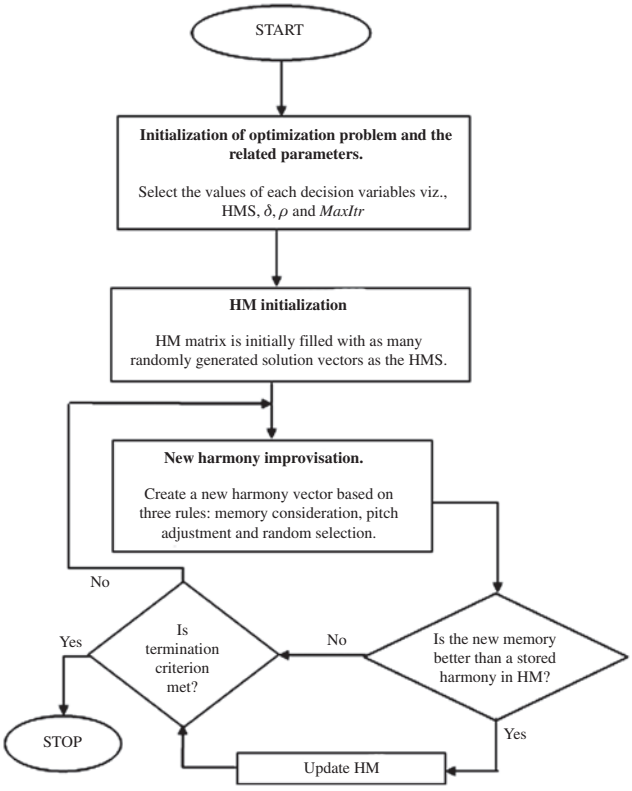

Figure 3: Flowchart of the HSA Procedure.

5.3 Example of FS Using HSA

All the data used in this example are hypothetical. Let us assume we have a total of 20 features, i.e. $f_1, f_2, f_3, \ldots \ldots f_{20}$, and we want to select the best five features among them. Thus, the total number of variables taken under consideration is five. Here, our objective function is the classification accuracy evaluated by a given machine learning algorithm. The goal is to maximize this classification accuracy using HSA. Let us consider the HMS to be 6.

	HMS	Classification Score
Set #1	$<f_1, f_5, f_3, f_2, f_9>$	40%
Set #2	$<f_2, f_7, f_4, f_1, f_6>$	60%
Set #3	$<f_{10}, f_1, f_2, f_3, f_7>$	70%
Set #4	$<f_{12}, f_{17}, f_3, f_{19}, f_1>$	50%
Set #5	$<f_1, f_{10}, f_{20}, f_{13}, f_5>$	30%
Set #6	$<f_2, f_9, f_5, f_{14}, f_{18}>$	63%

Here, each solution vector is generated randomly. Each musician must select a value from his or her note domain at the time of improvisation to generate a new solution vector. This selection of note, from the existing domain of notes of that particular musician, is controlled by two parameters (i.e. δ and ρ). If we set $\delta = 0.9$ and $\rho = 0.1$, then the first musician chooses a new note from the domain $\{f_1, f_2, f_{10}, f_{12}, f_1, f_2\}$ with 0.9 probability; however, there still exists a probability of 0.1, i.e. (1 – 0.9), that the first musician may choose from the range of all possible attribute values [i.e. $(f_1, f_2, f_3, \ldots \ldots .. f_{20})$]. Let us assume that the first musician has chosen f_{17} value from all possible existing note values; then, as per our HSA, the rest of the musicians cannot choose the f_{17} value. Thus, f_{17} needs to be excluded from the domain of the second musician; thus, the second musician can choose a new note from the domain $\{f_5, f_7, f_1, f_{10}, f_9\}$ with 0.9 probability as $\delta = 0.9$. Similarly, there exists a probability of 0.1 that the second musician may choose from the range of all possible attributes values [i.e. $(f_1, f_2, f_3, \ldots, f_{16}, f_{18}, f_{19}, f_{20})$]. This FS based on δ value may not be the final selection in the new solution vector, as ρ can shift the previously selected feature. For example, here, the first musician has already chosen f_{17}; however, as $\rho = 0.1$, after making these choices, the first musician can choose left or right neighbors of f_{17} with 0.1 probability. If $P_{band} = 2$, thus the first musician has total choices of five values $\{f_{15}, f_{16}, f_{17}, f_{18}, f_{19}\}$. Here, P_{band} indicates that the first musician can reach up to the position along the left or right neighboring direction. Let us assume, at the end, that the first musician has chosen f_{18}. In this way, all the musicians need to select note values. At the final stage of the improvisation process, let the new solution vector is $(f_{18}, f_1, f_{20}, f_{14}, f_4)$. Now, with this feature set, testing is done using any supervised classifier. Let us assume that we obtained 80% classification accuracy using the newly generated solution vector. This is better than the worst scored solution vector [i.e. $<f_1, f_{10}, f_{20}, f_{13}, f_5>$ with 30% classification accuracy] in the HM. Thus, we update the memory with the newly generated solution vector by replacing the worst solution vector. This type of improvisation process continues until *MaxItr* is met.

5.4 Application of HSA to Modified Log-Gabor Filter Technique

This subsection describes the selection of optimal features extracted from the modified log-Gabor filter procedure. Here, our main objective is to reduce the number of features using HSA so that the redundant or misleading features get eliminated, which, in turn, increases the classification accuracy. Initially, let us assume 20% of the total number of features (i.e. 120) would provide a good classification score. This statement is also validated by increasing the total number of features by 20% in the next experiment. In this way, the experimentation is done by taking 20%, 40%, 60%, and 80% of the original feature set. That is, the increment factor (IF) is set to 20%. Here, we have a total of 120 features, i.e. $f_1, f_2, f_3, \ldots \ldots .. f_{120}$, and suppose we want to select 24 optimal features among them (as our initial assumption is that 20% of total

feature can give us a good result). Thus, the total number of variables taken under consideration is 24. Let us consider the initial value of HMS to be 10. At a later stage, this value is made to vary, and it is found that the initial value is the sufficient value as an increase of HMS value does not affect the overall result in our experiment.

The pseudo code for selection of optimal feature subset for the proposed work is as follows.

Here, we have a total of 120 features, i.e. $f_1, f_2, f_3, \ldots \ldots .. f_{120}$, and initially we have tried with 24 features in order to find the optimal set of feature of length 24.

Here, 24 features are used to define the objective function $f(x)$. Let us consider we have features $f_i, f_j, f_k, \ldots \ldots .. f_m$, where the subscript of each feature denotes the corresponding feature number and $1<=i, j, k, \ldots . m<=120$ and $i \neq j \neq k \neq \ldots \ldots . \neq m$. Here, our objective function is the maximum classification accuracy evaluated by a given machine learning algorithm. The goal is to maximize this classification accuracy by applying HSA.

Begin

Initialize PAR and bandwidth
Initialize HMCR, HMS, IF, and *MaxItr*
Initialize the number of decision variable *v* (for this experiment, *v* is initialized to 20% of the original feature set)
Fill the HMS with some probable solution vectors by choosing random harmonies between f_1 and f_{120}
For $c = 1$ to ((100 / IF) – 1)
// loop will execute four times, i.e. 20%, 40%, 60%, and 80%, as IF = 20%
{

while ($t < MaxItr$)
//*t* indicating current iteration number must be initialized with 1

{
For each musician (x_i)
//here total number of musician depends on *v*

{
Generate a random number rn_1, where $0 \leq rn_1 \leq 1$
if ($rn_1 <$ HMCR)
{
Randomly choose a note from the existing domain of x_i
Generate a random number rn_2, where $0 \leq rn_2 \leq 1$
if ($rn_2 <$ PAR)
{
Select the neighboring value of already selected note, this range of choosing the neighboring value depends on *b* [here, *b* needs to be chosen randomly].
}
}
else
{
Randomly choose a value for x_i between (f_1, f_{120})
}
// If the current value of x_i is found to be same with the previous value, then we need to continue the same procedure until distinct feature values appear. This is due to the fact that the same feature cannot be present twice in the probable solution.
} // **end For**
Evaluate the classification score for newly generated solution vector.
If the newly generated solution vector is better than the worst solution vector in HMS, then replace the worst solution vector with newly generated solution vector.
$t = t + 1$
} // **end while**
Increment *v* by IF

} // **end For**

---

6 Results and Analysis

The experiments are carried out on 12 official scripts of India, *namely Bangla*, *Devanagari*, *Gujarati*, *Gurumukhi*, *Kannada*, *Malayalam*, *Manipuri*, *Oriya*, *Tamil*, *Telugu*, *Urdu*, and *Roman*. All the experiments are implemented in MATLAB 2010 (The Mathworks, Natick, MA, USA) under a Windows XP (Microsoft, Redmond, WA, USA) environment on an Intel Core2 Duo 2.4 GHz processor with 2 GB of RAM. A total dataset of 120 handwritten document pages (10 pages from each script) is collected from different people of varying age, sex, educational qualification, etc. These documents are digitized using an HP flatbed scanner and stored as 24-bit bitmap images. The gray-tone images may contain noisy pixels that have been removed by using Gaussian filter [14]. A set of 160, 640, and 2560 text blocks for each document of different scripts are prepared by applying second-level, third-level, and fourth-level quad-tree-based page segmentation approaches. Sample handwritten text block images of second-level decomposition for all the above-mentioned scripts are shown in Figure 4. A 3-fold cross-validation scheme is used for testing the present approach. That is, for the second-level decomposition approach, a total of 1280 text blocks are used for the training purpose and the remaining 640 text blocks are used for the testing purpose. Similarly, for third-level decomposition, 5120 text blocks are taken for the training and the remaining 2560 text blocks are used for testing the system. Again, for fourth-level decomposition, a set of 20,480 and 10,240 text blocks are utilized for training and testing the present system, respectively. The proposed approach is then evaluated using six different conventional classifiers,

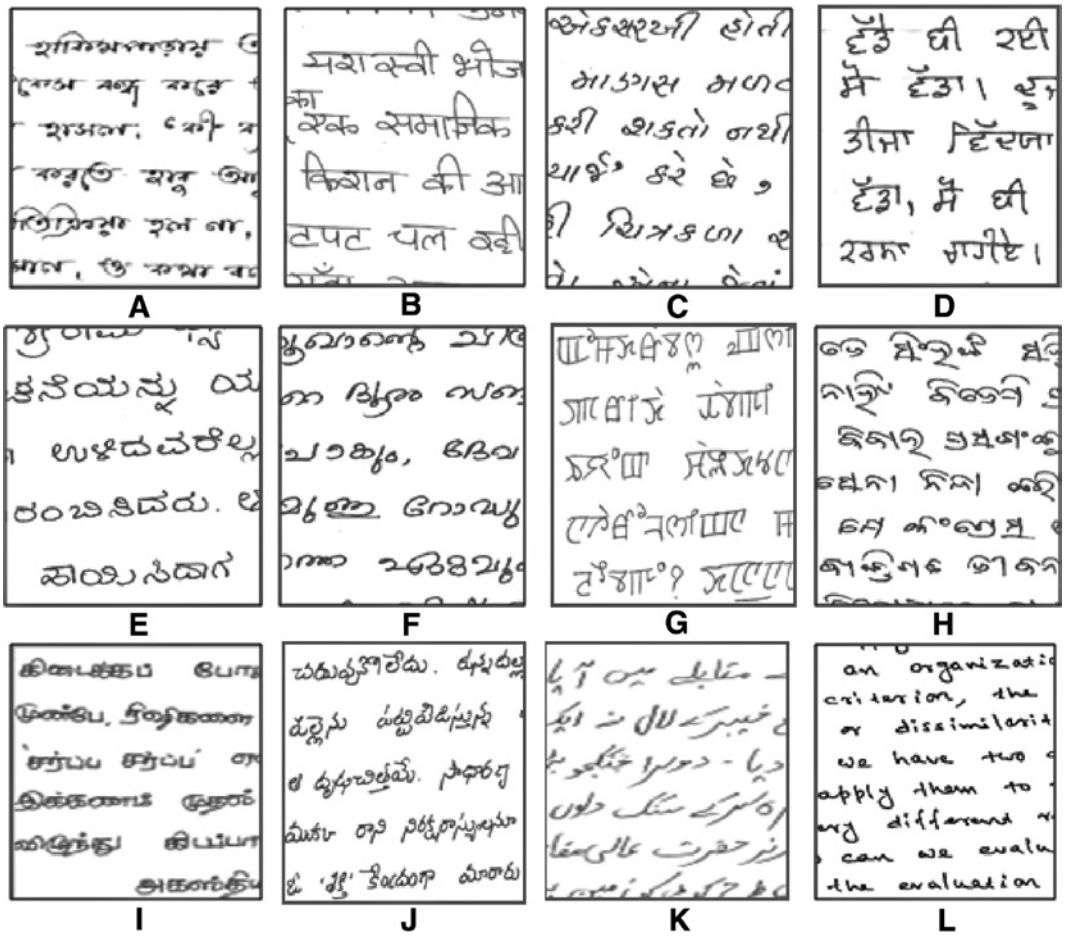

Figure 4: Samples of Text Block Images, Obtained at Second-Level Quad-Tree-Based Decomposition Scheme, Written in (A) *Bangla*, (B) *Devanagari*, (C) *Gujarati*, (D) *Gurumukhi*, (E) *Kannada*, (F) *Malayalam*, (G) *Manipuri*, (H) *Oriya*, (I) *Tamil*, (J) *Telugu*, (K) *Urdu*, and (L) *Roman* scripts.

namely Naïve Bayes, Multi Layer Perceptron (MLP), SVM, Random Forest, Bagging, MultiClass classifier. The graphical comparison of the identification accuracies of the said classifiers achieved with Modified log-Gabor filter-based features for second-level, third-level, and fourth-level page-segmentation approaches are shown with the help of a bar chart in Figure 5. It can be seen from the figure that the best identification accuracy is achieved by the SVM classifier, which is found to be 92.19%, 90.08%, and 89.26% for second-level, third-level, and fourth-level decomposition, respectively.

However, it is difficult to decide the optimum set of features that contributed positively in the above script identification experiments. Thus, the HS-based FS method is tested by selecting 20%, 40%, 60%, and 80% of the original feature vector (rounding off to the lowest nearest integer). Here, the objective function is the obviously the classification accuracy, which is evaluated as follows:

$$\text{Classification Accuracy} = \frac{\text{Number of correctly classified text blocks}}{\text{Total number of text blocks}} \times 100(\%). \tag{16}$$

As the best classification accuracy has been achieved using the SVM classifier, we have chosen the same for further experimentations. The SVM classifier effectively maps pattern vectors to a high-dimensional feature space where a "best" separating hyperplane (the "maximal margin" hyperplane) is constructed. To construct an optimal hyperplane, SVM employs an iterative training algorithm to minimize an error function, which can be defined as follows:

Find $W \in \mathbb{R}^m$ and $b \in \mathbb{R}$ to

$$\text{Minimize } \tfrac{1}{2} W^T W + C \sum_{i=1}^{N} \xi_i$$

subject to constraints $y_i(W^T \phi(x_i) + b) \geq 1 - \xi_i$, $i = 1, 2, \ldots\ldots, N$

$$\xi_i \geq 0, i = 1, 2\ldots\ldots, N,$$

where C is the penalty parameter of the error term, W is the vector of coefficients, b is a constant, and ξ_i represents parameters for handling non-separable data (inputs). The index i labels the N training cases. Note that $y \in \pm 1$ represents the class labels and x_i represents the independent variables. The kernel ϕ is used to transform data from the input (independent) to the feature space. It should be noted that the larger the C, the more the error is penalized. Thus, C should be chosen with care to avoid overfitting. There are number of kernels that can be used in SVM models. These include linear, polynomial, radial basis function, and sigmoid. For the present work, we have applied polynomial kernel, which can be written as

$$K(X_i, X_j) = (\gamma X_i . X_j + C)^d \tag{17}$$

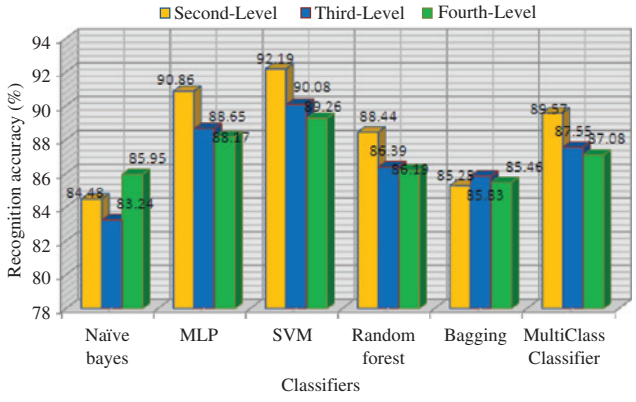

Figure 5: Graph Showing the Recognition Accuracies of Modified Log-Gabor Filter-Based Feature Extraction Method for Second-Level, Third-Level, and Fourth-Level Decomposition Approaches Using Six Different Classifiers.

where $K(X_i, X_j) = \phi(X_i).\phi(X_j)$ is the kernel function that represents a dot product of input data points mapped into the higher-dimensional feature space by transformation ϕ.

For the present work, the optimal values of δ and ρ have been experimentally set to 0.7 and 0.3, respectively. Keeping these values fixed, a comparison between the different values of HMS (HMS = 5, 10, 15, and 20) varying the number of iterations (*MaxItr* = 30, 50, and 70) and features (24, 48, 72, and 96) has been performed, which is detailed in Table 1. It can be observed from the table that for HMS = 10 and *MaxItr* = 50, the algorithm attains the highest recognition accuracies of 98.18%, 96.56%, and 94.38% for second-level, third-level, and

Table 1: Recognition Accuracies of the Present HSA for Different Parameter Values of HMS at (A) Second Level, (B) Third Level, and (C) Fourth Level (Best Case Is Shaded in Gray and Styled in Bold).

	Number of Optimal features selected	*MaxItr*	Classification accuracy (%)			
			HMS			
			5	10	15	20
(A) Second level	24	30	91.06	92.24	93.69	92.5
		50	91.59	93.18	93.35	92.33
		70	93.29	93.35	92.73	92.39
	48	30	97.01	96.95	97.17	97.17
		50	97.16	96.87	97.27	96.99
		70	96.99	97.39	97.27	97.21
	72	30	97.65	97.95	97.95	98.01
		50	97.95	**98.18**	97.95	98.01
		70	97.95	97.84	98.01	97.95
	96	30	98.01	98.07	98.12	98.12
		50	98.18	98.07	98.12	98.18
		70	98.12	98.18	98.12	98.18
(B) Third level	24	30	91.18	92.35	92.35	91.62
		50	91.18	92.12	92.02	91.62
		70	91.86	92.19	92.53	92.15
	48	30	94.66	94.52	95.10	94.98
		50	94.89	94.93	95.06	94.98
		70	95.07	95.17	95.10	95.04
	72	30	96.21	96.24	95.5	95.68
		50	96.21	**96.56**	95.39	95.37
		70	95.24	95.57	95.21	95.68
	96	30	95.68	96.24	95.68	95.68
		50	95.68	96.24	95.63	95.68
		70	95.68	95.75	95.68	95.68
(C) Fourth level	24	30	91.55	92.86	92.3	92.39
		50	91.55	91.06	92.3	91.18
		70	91.78	92.42	92.86	92.39
	48	30	92.55	93.08	93.08	93.25
		50	92.35	93.08	93.05	93.17
		70	93.19	94.21	93.68	93.25
	72	30	93.89	94.21	93.2	93.2
		50	93.56	**94.38**	93.2	93.15
		70	93.89	93.75	93.57	93.09
	96	30	93.59	93.68	93.98	93.25
		50	94.09	93.28	93.98	93.39
		70	93.59	93.67	93.25	93.25

fourth-level decomposition, respectively, which is significantly higher than the preceding outcomes. Apart from this, the optimal number of features required by the HSA is reduced to 72, which comprise almost 60% of the original feature set described in Section 3. In the present work, statistical performance measures with respect to different parameters, *namely* kappa statistics, mean absolute error, root mean square error, true positive rate, false positive rate, precision, recall, *F*-measure, and area under ROC are computed for 12 different scripts using the SVM classifier. The said measurements at the second, third, and fourth levels of page decomposition are detailed in Table 2A–C, respectively.

6.1 Sensitivity to Noise

In this subsection, the accuracy of feature subset selection using HSA has been measured by studying the impact of Gaussian noise to the handwritten script document images at all the three levels of decomposition. This is done to validate the robustness of the present methodology in the presence of noise. Presently, we have added 10%, 20%, and 30% Gaussian noises into the pages, and experimented at all the three levels of page decomposition. An example showing the addition of Gaussian noise on a sample handwritten *Kannada* image at second-level decomposition is illustrated in Figure 6. The Modified log-Gabor filter (as described in Section 3) is again applied to extract the original features, and the number of features is the optimized HSA. The values of parameters of HSA for which the highest identification accuracies have been obtained (in the previous subsection) are taken into consideration for the current experiment. The classification is done using the SVM classifier. A comparative study of the script identification accuracies achieved in the presence of noise at all the three levels of page decomposition are shown with the help of a bar chart in Figure 7. It is to be noted from the figure that the present methodology shows a considerable amount of resistivity to noise. The script recognition accuracies have been found to decrease only about 1%, 2%, and 3% than the present recognition accuracies for 10%, 20%, and 30% addition of Gaussian noise at second-, third-, and fourth-level decompositions, respectively. Thus, it can be concluded from this experiment that the present methodology is less sensitive to noise to a large extent.

6.2 Statistical Significance Tests

The statistical significance of the experimental setup has also been measured as an essential part for validating the performance of the multiple classifiers using multiple datasets. For statistical comparison of multiple classifiers, two or more learning algorithms are first trained and tested on suitable datasets and then evaluated using the classification accuracies. Datasets are randomly divided into small datasets with different sample sizes. Performances of different classifiers are then carried out for each randomly created dataset. The only requirement for performing non-parametric tests is that the compiled results provide reliable estimates of the performances of the classification algorithms on each dataset. In the usual experimental setups, these numbers come from cross-validation or from repeated stratified random splits onto training and testing datasets. The term “sample size” refers to the number of datasets used, not the number of training/testing samples drawn from each individual set. The sample size can therefore be as small as five and is usually well below 30. In the present work, we have performed some statistical significance tests for validating the results of the best classifier among six different classifiers at second-level decomposition. These tests have been performed at two stages: (a) before applying HSA and (b) after applying HSA. The theoretical discussions related to these tests are already described in one of our previous works [16].

Case I: Application of Statistical Significance Tests prior to HSA

A safe and robust non-parametric Friedman test [14] has been conducted along with the corresponding post-hoc tests. For the experimentation, the number of datasets (N) and the number of classifiers (k) are set as 12

Table 2: Statistical Performance Measures along with Their Respective Means (Styled in Bold and Shaded in Gray) Achieved for Best Case of the SVM Classifier on 12 Different Scripts at (A) Second-Level, (B) Third-Level, and (C) Four-Level Page Decomposition.

Script	Statistical performance measures								
	Kappa statistics	MAE	RMSE	TPR	FPR	Precision	Recall	*F*-measure	AUC
(A)									
Bangla	0.98	0.1621	0.2819	0.963	0.003	0.969	0.963	0.966	0.999
Devanagari				0.950	0.001	0.987	0.950	0.968	0.993
Gujarati				0.994	0.001	0.994	0.994	0.994	1.000
Gurumukhi				0.988	0.003	0.95	0.988	0.981	0.994
Kannada				1.000	0.003	0.976	1.000	0.988	1.000
Malayalam				1.000	0.001	0.988	1.000	0.994	1.000
Manipuri				0.982	0.002	0.982	0.982	0.982	0.980
Oriya				0.981	0.002	0.969	0.981	0.975	1.000
Tamil				0.981	0.002	0.981	0.981	0.981	0.999
Telugu				0.981	0.003	0.981	0.981	0.981	0.999
Urdu				0.981	0.001	0.98	0.981	0.984	0.999
Roman				0.981	0.001	0.994	0.981	0.987	1.000
Average	**0.98**	**0.1621**	**0.2819**	**0.982**	**0.002**	**0.982**	**0.982**	**0.982**	**0.998**
(B)									
Bangla	0.9622	0.0227	0.0805	0.978	0.003	0.968	0.978	0.973	0.998
Devanagari				0.936	0.002	0.979	0.936	0.957	0.993
Gujarati				0.980	0.005	0.949	0.980	0.964	0.995
Gurumukhi				0.947	0.004	0.963	0.947	0.955	0.994
Kannada				0.981	0.003	0.975	0.981	0.978	0.997
Malayalam				0.984	0.002	0.981	0.984	0.983	0.999
Manipuri				0.966	0.003	0.966	0.966	0.966	0.995
Oriya				0.955	0.003	0.974	0.955	0.964	0.994
Tamil				0.989	0.007	0.938	0.989	0.963	0.997
Telugu				0.922	0.007	0.932	0.922	0.927	0.982
Urdu				0.986	0.001	0.987	0.986	0.987	0.999
Roman				0.964	0.002	0.978	0.964	0.971	0.997
Average	**0.9622**	**0.0227**	**0.0805**	**0.966**	**0.003**	**0.966**	**0.966**	**0.966**	**0.995**
(C)									
Bangla	0.9382	0.0327	0.1018	0.986	0.007	0.937	0.986	0.961	0.994
Devanagari				0.861	0.006	0.938	0.861	0.898	0.968
Gujarati				0.965	0.007	0.933	0.965	0.949	0.990
Gurumukhi				0.918	0.005	0.950	0.918	0.934	0.983
Kannada				0.970	0.002	0.978	0.970	0.974	0.997
Malayalam				0.982	0.003	0.967	0.982	0.975	0.998
Manipuri				0.944	0.006	0.944	0.944	0.944	0.986
Oriya				0.948	0.004	0.959	0.948	0.953	0.990
Tamil				0.963	0.012	0.893	0.963	0.927	0.980
Telugu				0.882	0.010	0.902	0.882	0.892	0.961
Urdu				0.945	0.005	0.954	0.945	0.950	0.990
Roman				0.963	0.002	0.975	0.963	0.969	0.996
Average	**0.9382**	**0.0327**	**0.1018**	**0.944**	**0.006**	**0.944**	**0.944**	**0.944**	**0.986**

MAE, mean absolute error; RMSE, root mean square error; TPR, true positive rate; FPR, false positive rate; AUC, area under ROC.

and 6, respectively. These datasets are chosen randomly from the test set. The performances of the classifiers on different datasets are shown in Table 3. On the basis of these performances, the classifiers are then ranked for each dataset separately, the best-performing algorithm gets the rank 1, the second best gets the rank 2, and so on (see Table 3). Average ranks are assigned in case of ties.

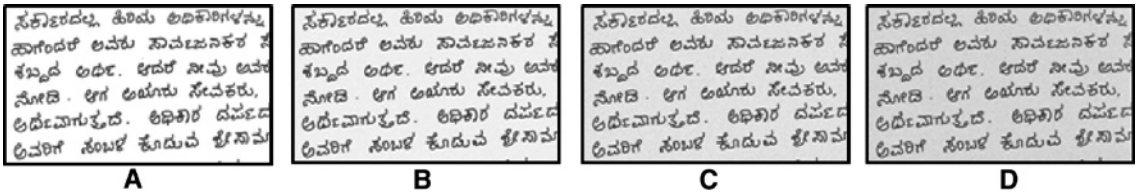

Figure 6: Illustration of (A) Original Handwritten *Kannada* Image at Second-Level Decomposition, and Addition of Gaussian Noise at (B) 10%, (C) 20%, and (D) 30%.

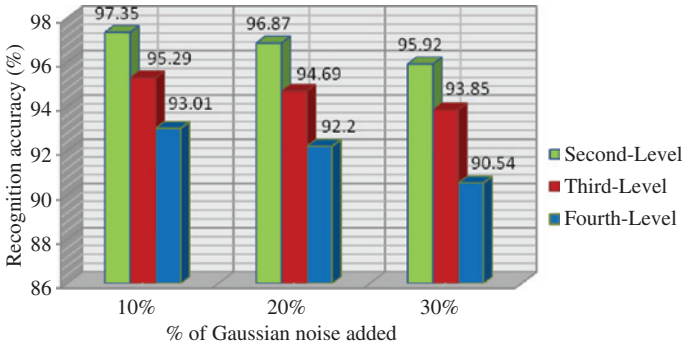

Figure 7: Comparison of the Impact of Added Gaussian Noise to Handwritten Script Identification Accuracies (Evaluated Using the SVM Classifier) at All Three Levels of Page Decomposition.

Table 3: Recognition Accuracies of Six Classifiers and Their Corresponding Ranks Using 12 Different Datasets Before Performing HSA (Ranks in Parentheses Are Used for Performing Friedman Test).

Dataset	Classifier					
	Naïve bayes (%)	MLP (%)	SVM (%)	Random forest (%)	Bagging (%)	Multiclass classifier (%)
#1	90 (5)	93 (3)	98 (1)	88 (6)	91 (4)	95 (2)
#2	88 (6)	93 (2)	96 (1)	91 (3)	90 (4)	89 (5)
#3	93 (3)	91 (4)	95 (2)	90 (5.5)	90 (5.5)	97 (1)
#4	91 (4.5)	95 (2)	97 (1)	93 (3)	90 (6)	91 (4.5)
#5	90 (6)	95 (2)	96 (1)	93 (4)	91 (5)	95 (3)
#6	96 (2)	90 (5.5)	98 (1)	90 (5.5)	91 (4)	93 (3)
#7	93 (2)	91 (5)	95 (1)	91 (5)	92 (3)	91 (5)
#8	88 (6)	95 (3)	96 (1.5)	92 (4)	90 (5)	96 (1.5)
#9	95 (2)	94 (3)	97 (1)	90 (6)	91 (4.5)	91 (4.5)
#10	92 (2)	86 (5.5)	94 (1)	90 (3)	86 (5.5)	88 (4)
#11	90 (4.5)	90 (4.5)	95 (1)	93 (2)	91 (3)	88 (6)
#12	89 (6)	94 (2.5)	96 (1)	94 (2.5)	90 (5)	92 (4)
Mean rank	**$R_1 = 4.08$**	**$R_2 = 3.5$**	**$R_3 = 1.125$**	**$R_4 = 4.125$**	**$R_5 = 4.54$**	**$R_6 = 3.625$**

The mean ranks scored by individual classifiers are shaded in gray and styled in bold.

Let r_j^i be the rank of the j-th classifier on i-th dataset. Then, the mean of the ranks of the j-th classifier over all the N datasets will be computed as

$$R_j = \frac{1}{N}\sum_{i=1}^{N} r_j^i \tag{18}$$

The null hypothesis states that all the classifiers are equivalent and so their ranks R_j should be equal. To justify it, the Friedman statistic [16] is computed as follows:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{19}$$

Under the current experimentation, this statistic is distributed according to χ_F^2 with $k-1$ (=5) degrees of freedom. Using Eq. (19), the value of χ_F^2 is calculated as 25.47. From the table of critical values (see any standard statistical book), the value of χ_F^2 with 5 degrees of freedom is 11.07 for $\alpha = 0.05$ (where α is known as the level of significance). It can be seen that the computed χ_F^2 differs significantly from the standard χ_F^2. Thus, the null hypothesis is rejected.

Iman et al. [16] have derived a superior statistic using the following formula:

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{20}$$

F_F is distributed according to the F-distribution with $k-1$ (=5) and $(k-1)(N-1)$ (=55) degrees of freedom. Using Eq. (20), the value of F_F is calculated as 8.115. The critical value of F (5, 55) for $\alpha = 0.05$ is 4.43 (see any standard statistical book), which shows a significant difference between the standard and calculated values of F_F. Thus, both the Friedman and Iman et al. statistics reject the null hypothesis.

As the null hypothesis is rejected, a post-hoc test known as the Nemenyi test [16] is carried out for pairwise comparisons of the best- and worst-performing classifiers. The performances of two classifiers are significantly different if the corresponding average ranks differ by at least the critical difference (*CD*), which is expressed as

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{21}$$

For Nemenyi's test, the value of $q_{0.05}$ for six classifiers is 2.850 (see Table 5a of Ref. [6]). Thus, the *CD* is calculated as $2.850\sqrt{\frac{6*7}{6*12}}$, i.e. 2.17 using Eq. (21). As the difference between the mean ranks of the best and worst classifier is much greater than the *CD*, we can conclude that there is a significant difference between the performing ability of the classifiers. For comparing all the classifiers with a control classifier (here SVM), we have applied the Bonferroni-Dunn test [16]. For this test, *CD* is calculated using the same Eq. (21). However, here, the value of $q_{0.05}$ for six classifiers is 2.576 (see Table 5b of Ref. [6]). Thus, the *CD* for the Bonferroni-Dunn test is calculated as $2.576\sqrt{\frac{6*7}{6*12}}$, i.e. 1.96. As the difference between the mean ranks of any classifier and SVM is always greater than *CD*, the chosen control classifier performs significantly better than other classifiers. A graphical representation of the said post-hoc tests for comparison of six different classifiers is shown in Figure 8.

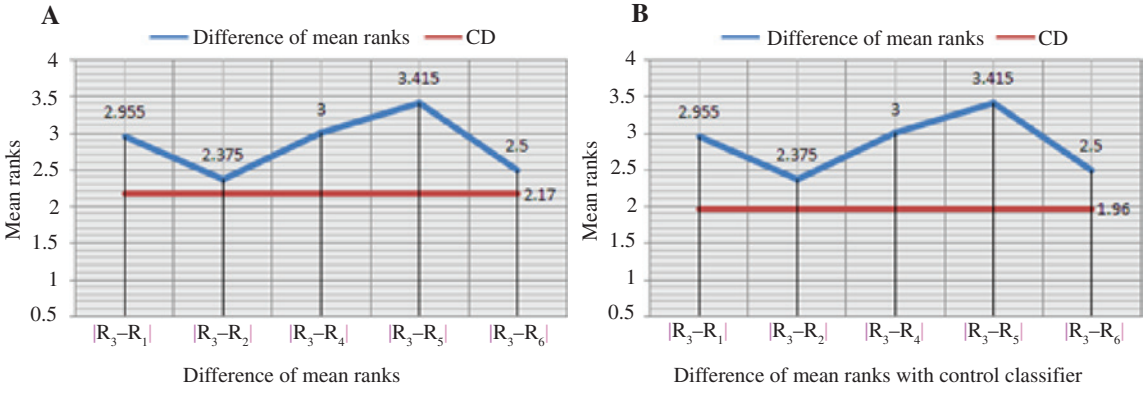

Figure 8: Graphical Representation of Comparison of Multiple Classifiers for (A) Nemenyi's Test and (B) Bonferroni-Dunn's Test Before Applying HSA.

After performing the above-mentioned statistical significance tests over the 12 datasets and six classifiers, it can be concluded that the SVM classifier outperforms all other classifiers at second-level decomposition. Similarly, the performances of these tests at the remaining two levels also confirm that the SVM classifier is found to be statistically significant than the other five classifiers.

Case II: Application of Statistical Significance Tests subsequent to HSA

Similar to the preceding case, the number of datasets (N) and the number of classifiers (k) are chosen as 12 and 6, respectively. The performances of the classifiers on different datasets are shown in Table 4.

Using the Friedman statistic given in Eq. (19), the value of χ_F^2 is calculated as 27.057. It can be seen that the computed χ_F^2 differs significantly from the standard χ_F^2. Using the Iman et al. statistic written in Eq. (20), the tabulated value of F_F is calculated as 9.035, which is much greater than the observed value of F_F for (5,55) degrees of freedom. Thus, for both tests, the null hypothesis is rejected.

For Nemenyi's test, the value of CD is calculated as $2.850\sqrt{\frac{6*7}{6*12}}$, i.e. 2.17, using the same Eq. (21). As the difference between the mean ranks of the best and worst classifier is much greater than the CD, we can conclude that there is a significant difference between the performing ability of the classifiers. For comparing all the classifiers with a control classifier using the Bonferroni-Dunn test, the value of CD is calculated as $2.576\sqrt{\frac{6*7}{6*12}}$, i.e. 1.96. As the difference between the mean ranks of any classifier and SVM is always greater than CD, the chosen control classifier performs significantly better than other classifiers. A graphical representation of the above-mentioned post-hoc tests for comparison of six different classifiers is shown in Figure 9.

After performing the above-mentioned statistical significance tests over the 12 datasets and six classifiers, it can be concluded that the SVM classifier outperforms all other classifiers at second-level decomposition. Similarly, the performance of these tests at the remaining two levels also confirms that the SVM classifier is found to be statistically significant than the other five classifiers.

6.3 Comparison with Popular Meta-heuristic Algorithms

For the present work, the optimization performance of HSA is also compared with four popular meta-heuristic algorithms, *namely* GA [15], TS [36], PSO [19], and ACO [8]. The GA is implemented to contain 20 chromo-

Table 4: Recognition Accuracies of Six Classifiers and Their Corresponding Ranks Using 12 Different Datasets after Applying HSA (Ranks in Parentheses Are Used for Performing Friedman Test).

Dataset	Classifier					
	Naïve bayes (%)	MLP (%)	SVM (%)	Random forest (%)	Bagging (%)	Multiclass classifier (%)
#1	93 (6)	99 (2.5)	100 (1)	97 (5)	99 (2.5)	98 (4)
#2	99 (2.5)	99 (2.5)	100 (1)	96 (5.5)	96 (5.5)	97 (4)
#3	98 (4.5)	93 (6)	100 (1)	99 (2.5)	98 (4.5)	99 (2.5)
#4	93 (6)	98 (3)	99 (1.5)	99 (1.5)	97 (4)	96 (5)
#5	95 (6)	99 (2.5)	100 (1)	99 (2.5)	97 (5)	98 (4)
#6	99 (2.5)	96 (6)	100 (1)	97 (5)	98 (4)	99 (2.5)
#7	91 (6)	99 (3)	100 (1)	99 (3)	99 (3)	98 (5)
#8	93 (6)	98 (3.5)	100 (1)	98 (3.5)	96 (5)	99 (2)
#9	95 (6)	100 (1.5)	100 (1.5)	97 (5)	99 (3.5)	99 (3.5)
#10	99 (2.5)	99 (2.5)	100 (1)	97 (5.5)	97 (5.5)	98 (4)
#11	94 (6)	98 (4.5)	100 (1)	99 (2.5)	98 (4.5)	99 (2.5)
#12	98 (3.5)	97 (5.5)	100 (1)	98 (3.5)	99 (2)	97 (5.5)
Mean rank	**$R_1 = 4.79$**	**$R_2 = 3.58$**	**$R_3 = 1.083$**	**$R_4 = 3.75$**	**$R_5 = 4.08$**	**$R_6 = 3.708$**

The mean ranks scored by individual classifiers are shaded in gray and styled in bold.

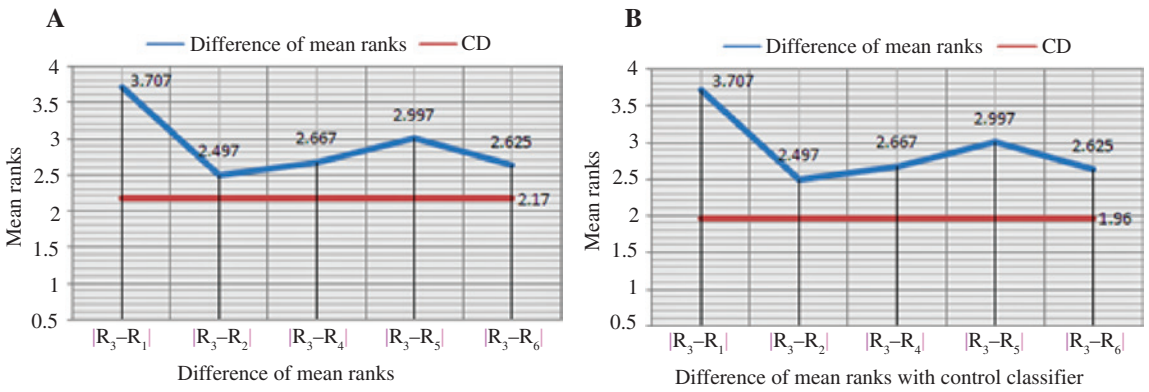

Figure 9: Graphical Representation of Comparison of Multiple Classifiers for (A) Nemenyi's Test and (B) Bonferroni-Dunn's Test After Applying HSA.

somes and made to run for 100 generations in each trial. The crossover rate and mutation rate are set to 1.0 and 0.1, respectively. The TS algorithm contains a list size of 10 tabus with a neighborhood size of 20, and this algorithm is made to run for 100 iterations. For the PSO algorithm, the number of particles used is 20. The two factors $rand_1$ and $rand_2$ are random numbers between (0, 1), whereas c_1 and c_2 are acceleration (learning) factors, with $c_1 = c_2 = 2$. The inertia weight w is taken as 0.8, and the maximum number of iterations used in our PSO is 100. Lastly, the ACO algorithm uses the following parameter values: (a) maximum number of iterations = 100, (b) evaporation rate = 0.3, and (c) maximum local search = 10. The comparison is measured with respect to three parameters: (a) optimal feature subset, (b) classification accuracy, and (c) execution time. The optimal feature subset denotes the optimal number of features required to achieve the maximum recognition accuracy. The execution time is measured as the time required to run the given optimization algorithm until the corresponding classification accuracy is realized using the SVM classifier with 3-fold cross-validation scheme. The comparison enlisted in Table 5 confirms that the present HSA not only achieves the highest classification accuracy but also involves the minimum computational time. Again, two well-known statistical feature dimensionality reduction techniques, named PCA [28] and greedy attribute search (GAS) [37], are also implemented and evaluated on the same dataset. From the experiments (shown in Table 6), it is found that both PCA and GAS utilize a feature dimensionality of 70 yielding recognition accuracies of 94.58%, 93.97%,

Table 5: Comparison of HSA with the Best Case of the Optimization Performances of GA, TS, PSO, and ACO Algorithms.

Optimization algorithms	Page segmentation level	Optimum feature subset	Classification accuracy (%)	Execution time (s)
GA [15]	2nd	**80**	92.32	1506.85
	3rd		93.54	2059.68
	4th		91.02	2945.36
TS [36]	2nd	**80**	92.65	1521.35
	3rd		93.76	2038.96
	4th		92.45	2852.28
PSO [19]	2nd	**75**	93.55	1355.29
	3rd		93.98	1968.03
	4th		90.46	2571.54
ACO [8]	2nd	**75**	95.26	1425.36
	3rd		92.85	2098.22
	4th		91.64	2464.55
HS	**2nd**	**72**	**98.18**	**1203.49**
	3rd		**96.56**	**1755.75**
	4th		**94.38**	**2236.20**

Highest recognition accuracy is shaded in gray and styled in bold.

Table 6: Comparison of HSA with PCA and GAS Techniques.

Statistical feature dimensionality reduction techniques	Page segmentation level	Optimum feature subset	Classification accuracy (%)	Execution time (s)
PCA [27]	2nd	**70**	94.58	1115.95
	3rd		93.97	1568.34
	4th		92.34	2055.6
GAS [37]	2nd	**70**	93.7	1328.55
	3rd		94.01	1945.27
	4th		90.25	2385.09
HS	**2nd**	**72**	**98.18**	**1203.49**
	3rd		**96.56**	**1755.75**
	4th		**94.38**	**2236.20**

Highest recognition accuracy is shaded in gray and styled in bold.

92.34% and 93.7%, 94.01%, 90.25% at second, third, and fourth levels of page segmentations, respectively. Although PCA selects fewer number of features as well as requires less computational time than HSA, HSA achieves almost about 4%, 3%, and 2% more recognition accuracies than the PCA at second-, third-, and fourth-level decomposition, respectively. Therefore, it is rather clear from the preceding observations that the present HSA outperforms some previously pioneered optimization algorithms used mainly for FS.

7 Conclusion

This paper has investigated the HSA for the selection of the optimal feature subset in handwritten script identification problem. The proposed HSA evolved upon the HM, which adaptively stores updated solutions during the evolution. This algorithm has been evaluated on a limited dataset of script blocks written in 12 official Indian scripts. For obtaining the optimal classification accuracy, we have conducted a series of experiments to test the efficiency of HSA with different parameter settings. The proposed HSA achieves the highest classification accuracy of 98.18% at the second-level block decomposition approach, which is almost about 6% more than the earlier attained result while only considering almost 60% of the original feature set. The improved result justifies the need for selecting the subset of designed features for this type of pattern classification problem, where both local and global features are computed without knowing the exact importance of the features from all regions of a particular script block image. Although the results show a rather encouraging trend, much work can still be done to further increase the potential of the present work.

In particular, investigations regarding how the algorithm parameters may be better tuned are necessary. It would be useful to develop a better stopping criterion, which will, in turn, lessen the computation time. In addition, a method may be devised to dynamically specify the size of the HM, which affects the convergence rate and search parameters.

Acknowledgments: The authors are thankful to the Center for Microprocessor Application for Training Education and Research and Project on Storage Retrieval and Understanding of Video for Multimedia of Computer Science and Engineering Department, Jadavpur University, for providing infrastructure facilities during the progress of the work. The current work reported here was partially funded by University with Potential for Excellence, Phase-II, UGC, Government of India.

Bibliography

[1] M. H. Aghdam, N. Ghasem-Aghaee and M. E. Basiri, Text feature selection using ant colony optimization, *Expert Syst. Appl.* **36** (2009), 6843–6853.

[2] I. Ahmad, Feature selection using particle swarm optimization in intrusion detection, *Int. J. Distrib. Sensor Netw.* **9** (2015), 1–8.
[3] P. Bermejo, J. Gámez and J. Puerta, A GRASP algorithm for fast hybrid (filter-wrapper) feature subset selection in high dimensional datasets, *Pattern Recogn. Lett.* **32** (2011), 701–711.
[4] N. Das, R. Sarkar, S. Basu, M. Kundu and M. Nasipuri, A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application, *Appl. Soft Comput.* **12** (2012), 1592–1606.
[5] C. De Stefano, F. Fontanella, C. Marrocco and A. Scotto di Freca, AGA-based feature selection approach with an application to handwritten character recognition, *Pattern Recogn. Lett.* **35** (2014), 130–141.
[6] J. Demsar, Statistical comparisons of classifiers over multiple data sets, *J. Mach. Learn. Res.* **7** (2006), 1–30.
[7] S. Ding, Feature selection based F-score and ACO algorithm in support vector machine, in: *Proc. of 2nd International Symposium on Knowledge Acquisition and Modeling*, 2009.
[8] M. Dorigo, The ant system: optimization by a colony of cooperating agents, *IEEE Trans. Syst. Man Cybern. B* **26** (1996), 1–13.
[9] M. El Alami, A filter model for feature subset selection based on genetic algorithm, *Knowl. Based Syst.* **22** (2009), 356–362.
[10] D. J. Field, Relations between the statistics of natural images and the response properties of cortical cells, *J. Opt. Soc. Am. A* **4** (1987), 2379–2394.
[11] R. Forsati, A. Moayedikia and A. Keikha, A novel approach for feature election based on the bee colony optimization, *Int. J. Comput. Appl.* **43** (2012), 13–16.
[12] Z. W. Geem, Optimal cost design of water distribution networks using harmony search, Dissertation, Korea University, 2000.
[13] Z. W. Geem, J. H. Kim and G. V. Loganathan, A new heuristic optimization algorithm: harmony search, *Simulation* **76** (2001), 60–68.
[14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, vol. I. Prentice-Hall, India, 1992.
[15] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, New York, 1975.
[16] R. L. Iman and J. M. Davenport, Approximations of the critical region of the Friedman statistic, in: *Communications in Statistics*, pp. 571–595, 1980.
[17] M. Kabir, M. Islam and K. Murase, A new local search based hybrid genetic algorithm for feature selection, *Neurocomputing* **74** (2011), 2914–2928.
[18] S. Kashef and H. Nezamabadi-pour, An advanced ACO algorithm for feature subset selection, *Neurocomputing* **147** (2015), 271–279.
[19] J. Kennedy and R. C. Eberhart, Particle swarm optimization, in: *Proc. of IEEE International Conference on Neural Networks*, pp. 1942–1948, 1995.
[20] V. Kumar, J. K. Chhabra and D. Kumar, Automatic unsupervised feature selection using gravitational search algorithm, *IETE J. Res.* **61** (2014), 22–31.
[21] V. Kumar, J. K. Chhabra and D. Kumar, Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems, *J. Comput. Sci.* **5** (2014), 144–155.
[22] V. Kumar, J. K. Chhabra and D. Kumar, Variance based harmony search algorithm for unimodal and multimodal optimization problems with application to clustering, *Cybernet. Syst.* **45** (2014), 486–511.
[23] V. Kumar, J. K. Chhabra and D. Kumar, Automatic data clustering using parameter adaptive harmony search algorithm and its application to image segmentation, *J. Intell. Syst.* **25** (2015), 595–610.
[24] K. S. Lee and Z. W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, *Comput. Methods Appl. Mech. Eng.* **194** (2004), 3902–3933.
[25] S. W. Lin, K. C. Ying, S. C. Chen and Z. J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, *Expert Syst. Appl.* **35** (2008), 1817–1824.
[26] F. G. Mohammadi and M. S. Abadeh, Imagesteg analysis using a bee colony based feature selection algorithm, *Eng. Appl. Artif. Intell.* **31** (2014), 35–43.
[27] M. Nazir, A. Majid-Mirza and S. Ali-Khan, PSO-GA based optimized feature selection using facial and clothing information for gender classification, *J. Appl. Res. Technol.* **12** (2014), 145–152.
[28] K. Pearson, On lines and planes of closest fit to systems of points in space, *Philos. Mag.* **2** (1901), 559–572.
[29] A. Roy, N. Das, R. Sarkar, S. Basu, M. Kundu and M. Nasipuri, Region selection in handwritten character recognition using artificial bee colony optimization, in: *Proc. of 3rd International Conference on Emerging Applications of Information Technology (EAIT)*, pp. 183–186, 2012.
[30] A. Roy, N. Das, R. Sarkar, S. Basu, M. Kundu and M. Nasipuri, An axiomatic fuzzy set theory based feature selection methodology for handwritten numeral recognition, in: *Proc. of 48th Annual Convention of Computer Society of India*, vol. I, AISC248, pp. 133–140, 2014.
[31] A. K. Saxena and V. K. Dubey, A survey on feature selection algorithms, *Int. J. Recent Innov. Trends Comput. Commun.* **3** (2015), 1895–1899.
[32] D. A. A. G. Singh, E. Jebamalar Leavline, K. Valliyappan and M. Srinivasan, Enhancing the performance of classifier using particle swarm optimization (PSO)-based dimensionality reduction, *Int. J. Energ. Inform. Commun.* **6** (2015), 19–26.
[33] P. K. Singh, I. Chatterjee and R. Sarkar, Page-level handwritten script identification using modified log-Gabor filter based features, in: *Proc. of 2nd IEEE International Conference on Recent Trends in Information Systems (ReTIS)*, pp. 225–230, 2015.

[34] P. K. Singh, R. Sarkar and M. Nasipuri, Offline script identification from multilingual Indic-script documents: a state-of-the-art, *Comput. Sci. Rev.* **15–16** (2015), 1–28.
[35] R. Sivagaminathan and S. Ramakrishnan, A hybrid approach for feature subset selection using neural networks and ant colony optimization, *Expert Syst. Appl.* **33** (2007), 49–60.
[36] C. S. Sung and H. W. Jin, A tabu-search-based heuristic for clustering, *Pattern Recogn.* **33** (2000), 849–858.
[37] H. Vafaie and I. F. Imam, Feature selection methods: genetic algorithms vs. greedy-like search, in: *Proc. of 3rd International Fuzzy Systems and Intelligent Control Conference*, Louisville, KY, March 1994.
[38] V. M. Vanishree, Provision for linguistic diversity and linguistic minorities in India, Available at: http://www.languageinindia.com/feb2011/vanishreemastersfinal.pdf. Retrieved 5 February, 2016.
[39] S. Vieira, L. Mendonc, G. Farinha and J. Sousa, Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients, *Appl. Soft Comput.* **13** (2013), 3494–3504.
[40] L. T. Vinh, S. Lee, Y. T. Park and B. J. d'Auriol, A novel feature selection method based on normalized mutual information, *Appl. Intell.* **37** (2010), 100–120.
[41] Z. Zeng, H. Zhang, R. Zhang and Y. Zhang, A hybrid feature selection method based on rough conditional mutual information and naive Bayesian classifier, *ISRN Appl. Math.* (2014), 1–11. Article ID: 382738.