

# Self-adjusting harmony search-based feature selection

Ling Zheng · Ren Diao · Qiang Shen

© Springer-Verlag Berlin Heidelberg 2014

**Abstract** Many strategies have been exploited for the task of feature selection, 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. In particular, harmony search is a recently developed technique mimicking musicians' experience, which has been effectively utilised to cope with feature selection problems. In this paper, a self-adjusting approach is proposed for feature selection with an aim to further enhance the performance of the existing harmony search-based method. This novel approach includes three dynamic strategies: restricted feature domain, harmony memory consolidation, and pitch adjustment. Systematic experimental evaluations using high dimensional, real-valued benchmark data sets are conducted in order to verify the efficacy of the proposed work.

**Keywords** Feature selection · Harmony search · Harmony memory consolidation · Pitch adjustment strategy

## 1 Introduction

Feature selection (FS) (Guyon and Elisseeff 2003) is a concept to maximise or preserve a predictive outcome while minimising the dimensionality of given input features (Liu and Motoda 2008). In real-world applications, data often carries many features, which do not necessarily contain any useful information for the problem at hand. When analysing examples in a high-dimensional data problem (e.g., genomic micro-array) (Shah et al. 2012; Xing et al. 2001), the performance of a learned predictor becomes weak due to the high inter-dependency amongst individual features, or of combined subsets of features. This is the so-called “curse-of-dimensionality” problem (Bellman 1957). Applications such as text processing, data classification, and systems control (Mac et al. 2010; Shang and Barnes 2013; Shen and Jensen 2004) can benefit from the application of FS, after the noisy, irrelevant, redundant or misleading features being removed or reduced (Jensen and Shen 2009).

Given a data set with  $n$  dimensions, FS attempts to search for an “optimal” feature subset amongst  $2^n$  candidate subsets. An exhaustive method may be employed. However, it is often impractical for most data sets. Alternative approaches based on hill-climbing (HC) have been exploited where features are removed or added iteratively until there is no further improvement to the current candidate solution. These techniques may converge quickly, but their work on an individual feature basis may lead to a strong assumption that features are entirely independent of each other. Correlated features may potentially lose their required attention, whilst such inter-feature dependences are usually very common in real-world data. Thereby, HC approaches may result in the discovery of sub-optimal solution, both in terms of the quality evaluation score and the subset size. Alternative methods use random search or heuristic strategies in an attempt to over-

---

Communicated by Y. Jin.

---

L. Zheng · R. Diao · Q. Shen (✉)  
Department of Computer Science, Institute of Mathematics,  
Physics and Computer Science, Aberystwyth University,  
Aberystwyth, UK  
e-mail: qqs@aber.ac.uk

L. Zheng  
e-mail: liz5@aber.ac.uk

R. Diao  
e-mail: rrd09@aber.ac.uk

come such shortcomings. Nature-inspired heuristics such as genetic algorithms (GAs) (Grefenstette 1986), genetic programming (Muni et al. 2006), particle swarm optimisation (PSO) (Wang et al. 2007), simulated annealing (Aarts and Laarhoven 1989), and tabu search (Mashinchi et al. 2011) are also employed for FS with varying degrees of success.

Harmony search (HS) (Geem 2010) is a recently developed meta-heuristic optimisation algorithm mimicking musical improvisation phenomenon, during which, each musician plays a note for discovering a best harmony all together. HS has been very successful in performing various engineering optimisation problems (Betar et al. 2012; Das et al. 2011; Fesanghary et al. 2008; Lee and Geem 2004; Srinivasa et al. 2011; Vasebi et al. 2007; Zhang and Hanzo 2009) and machine learning tasks (Das et al. 2010; Mahdavi et al. 2008). Several advantages over traditional optimisation techniques have also been demonstrated (Geem 2010; Yang 2009). HS imposes only limited mathematical requirements and is not sensitive to the initial value settings. Being a population-based approach, HS works by generating a new vector that encodes a candidate solution, after considering the quality of existing tentative solutions. This is in contrast to the classical genetic algorithms that typically consider only two (parent) vectors in order to produce a new (child) vector. The original HS technique has been improved by methods that dynamically adjust its parameters (Geem 2010; Mahdavi et al. 2007), making the algorithm more adaptive to the variance in variable value ranges. Work has also been carried out to analyse the evolution of the population variance over successive generations in HS, thereby drawing important conclusions regarding its exploration power (Das et al. 2011).

An application of HS to FS (HSFS) has also been recently developed (Diao and Shen 2012), which has demonstrated competitive FS outcomes. However, the original HSFS is inflexible at adjusting the size of the parameter musician population, which directly affects the performance on feature subset size reduction. This weakness is alleviated to a certain extent by its iterative refinement extension, but the fundamental issue remains. Stochastic mechanisms have not been explored to their maximum potential by the original work, as it does not employ the parameter of pitch adjustment rate due to its ineffective mapping of concepts.

In this paper, a self-adjusting HSFS method is proposed. This new technique extends the original idea of HSFS, built on the basis of an earlier, preliminary investigation into the self-configuration of its internal components (Zheng et al. 2013), supported with substantial rigorous experimental evaluation. In particular, the concept of a restricted feature domain is introduced in order to limit the locally explorable solution domains (of individual musicians), allowing more informative features to be located more quickly, whilst also reducing the run-time memory requirement of the algorithm. A harmony memory consolidation mechanism is developed,

which allows musicians (that act as individual feature selectors in the algorithm) to exchange information on tentatively selected features locally, and helps identify and remove non-contributing musicians. As a result, the size of the musician group can be dynamically adjusted during the search. Furthermore, the pitch adjustment strategy is presented which mimics the pitch adjustment behaviour of instrumentalists. It is used by HSFS for fine tuning the emerging feature subsets. In such a scheme, a feature may be substituted by its neighbour, which is determined via the use of a certain feature similarity measure.

The remainder of this paper is organised as follows. Section 2 introduces the original HSFS algorithm that this work aims to improve upon. The proposed self-adjusting HSFS algorithm is described in Sect. 3, which includes the three newly developed techniques: restricted feature domain, harmony memory consolidation, and pitch adjustment. Section 4 provides results of a series of experimentations, which are carefully conducted for the purpose of demonstrating the benefits and characteristics of the presented strategies. These experimentations include comparative studies with alternative feature selection methods, tested over both typical benchmark datasets and real-world high-dimensional datasets. Finally, Sect. 5 summarises the paper and suggests the possible directions where the work may be refined and applied.

## 2 Feature selection with harmony search

HS (Geem 2010) is a meta-heuristic algorithm that attempts to find a solution vector that optimises a given (possibly multivariate) objective function. During such an iterative search process, each musician (decision variable) chooses a note (variable value) in order to find a best harmony (a potential global optimum) in conjunction with the other musicians. HS has a novel stochastic derivative (for discrete variables) based on musician's experience, rather than gradient (for continuous variables) in differential calculus. In this section, a general overview of the FS concepts is given, and the HS-based FS technique (HSFS) (Diao and Shen 2012) is also summarised for completeness.

An information system in the context of FS is a tuple  $\langle X, Y \rangle$ , where  $X$  is a non-empty set of finite objects, also referred to as the universe of discourse; and  $Y$  is a non-empty, finite set of features. For decision systems,  $Y = \{A \cup Z\}$  where  $A = \{a_1, \dots, a_{|A|}\}$  is the set of input features, and  $|A|$  denotes the cardinality of  $A$ , which may be either discrete- or real-valued; and  $Z$  is the set of decision features. Various methods have been developed in the literature for the purpose of evaluating the quality of a given feature subset  $B \subseteq A$ . The focus of this paper lies with the use of group-based feature subset evaluation (Dash and Liu 2003;

**Table 1** Concept mapping from HS to FS

HS	Optimisation	FS
Musician	Variable	Feature selector
Musician note	Variable value	Feature
Harmony	Solution vector	Subset
Harmony memory	Solution storage	Subset storage
Harmony evaluation	Fitness function	Subset evaluation
Optimal harmony	Optimal solution	Optimal subset

Hall 1998; Jensen and Shen 2009), which may particularly benefit from the stochastic properties of HSFS (Diao and Shen 2012). Such evaluation methods typically produce a normalised score  $f(B) \in [0, 1]$ ,  $f(\emptyset) = 0$ , for a given feature subset  $B$ , which is hereafter referred to as the feature subset evaluation score. Here  $f: \mathbb{B} \rightarrow \mathbb{R}$  represents a subset evaluation function that maps a set of feature subsets onto the set of real numbers.

## 2.1 Mapping of key notions

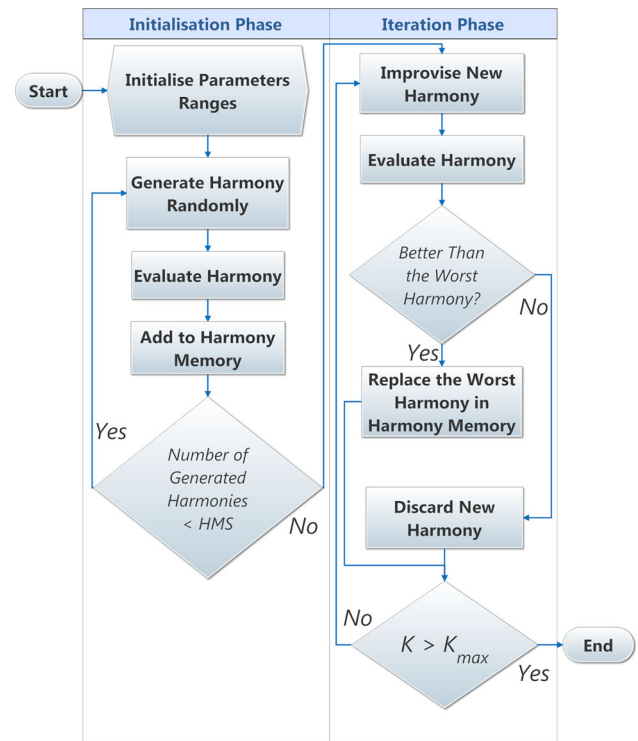
For conventional optimisation problems, the number of variables is pre-determined by the objective function to be optimised. However for HS, there is not a fixed number of elements in any potential candidate feature subset. In fact, the size of the emerging subset itself should be reduced in parallel to the optimisation of the subset quality evaluation score (Dash and Liu 2003; Hall 1998; Tsang et al. 2008). Therefore, when converting concepts, such as those shown in Table 1, a musician is best described as an independent expert or a “single feature selector”, where the available features translate to musical notes for musicians. Each musician may select one feature to be included in the emerging feature subset (the harmony), which is the combined vote from all musicians, indicating which features are being nominated.

The pool of the original features  $A$  forms the range of musical notes available to each of the musicians. Multiple musicians are allowed to choose the same feature, and they may opt to choose none at all. A feature subset quality evaluation method assesses each of the new subsets found during the search process. The original HSFS algorithm uses 4 parameters: HMS  $|\mathbb{H}|$ , the maximum number of iterations  $k_{\max}$ , the size of the musician group  $|M|$ , and the *harmony memory considering rate*  $\delta$  that encourages a musician to randomly choose from all available features (instead of within its own note domain).

Table 2 depicts the following three example harmonies.  $H^1$  denotes a subset of six distinctive features:  $B_{H^1} = \{a_1, a_2, a_3, a_4, a_7, a_{10}\}$ .  $H^2$  shows a duplication of choices from the first three musicians, and a discarded note (represented by  $a_-$ ) from musician  $m_6$ , representing a reduced subset  $B_{H^2} =$

**Table 2** Feature subsets encoding scheme

	$m_1$	$m_2$	$m_3$	$m_4$	$m_5$	$m_6$	Represented subset $B$
$H^1$	$a_2$	$a_1$	$a_3$	$a_4$	$a_7$	$a_{10}$	$\{a_1, a_2, a_3, a_4, a_7, a_{10}\}$
$H^2$	$a_2$	$a_2$	$a_2$	$a_3$	$a_{13}$	$a_-$	$\{a_2, a_3, a_{13}\}$
$H^3$	$a_2$	$a_-$	$a_3 \rightarrow a_6$	$a_2$	$a_{13}$	$a_4$	$\{a_2, a_4, a_6, a_{13}\}$

**Fig. 1** Proposed self-adjusting HSFS algorithm

$\{a_2, a_3, a_{13}\}$ .  $H^3$  signifies the feature subset  $B_{H^3} = \{a_2, a_4, a_6, a_{13}\}$ , where  $a_3 \rightarrow a_6$  indicates that  $m_4$  originally voted for  $a_3$ , but was forced to change the choice to  $a_6$  due to  $\delta$  activation. For simplicity, the explicit encoding/decoding process between a given harmony  $H^j$  and its associated feature subset  $B_{H^j}$  is omitted in the following explanation.

For conventional optimisation problems, the range of possible note choices for each musician is in general different from those for the other musicians. However, when applied to FS, all musicians jointly share one single value range, which is the set of all original features.

## 2.2 Work flow of HSFS

The iterative steps of the HSFS algorithm are shown in Fig. 1, where the operations of the approach are outlined.

- Initialise harmony memory

The parameters are assigned according to the given problem, including:  $|\mathbb{H}|$ ,  $|M|$ ,  $k_{\max}$ , and  $\delta$ . The subset stor-

age containing  $|\mathbb{H}|$  randomly generated subsets is then initialised. This provides each feature selector with a note domain of  $|\mathbb{H}|$  features, which may include identical choices and nulls.

- Improvise new subset

A new feature is chosen randomly by each feature selector out of their working feature domain, and such chosen features together form a new feature subset. In the event of  $\delta$  activation, a random feature will be chosen from all available features to substitute the feature selectors' own choice.

- Update subset storage

If the newly obtained subset achieves a higher evaluation score than that of the worst subset in the subset storage, the new subset is recorded and included in the subset storage and the existing worst subset is removed, otherwise this feature subset is discarded. The comparison of subsets takes into consideration of both evaluation score and subset size in order to discover the minimal reduction at termination.

- Iteration

The improvisation and comparative update procedure continues until a predefined maximum number of iterations  $k_{\max}$  is reached. The final output is the feature subset with the highest quality, out of those stored within the harmony memory at termination.

HSFS offers a clear advantage in that a group of features are evaluated as a whole. A newly improvised subset is not necessarily included in the subset storage, simply because one of the features has a locally strong evaluation score. This is the key distinction to any of the hill-climbing based approaches. The integer-valued subset representation also provides a greater freedom of diversification, allowing the stochastic mechanisms of HS to be better exploited. This approach enables the stochastic mechanisms offered by HS to be better exploited, allowing HSFS to achieve a higher search performance than its binary-valued counterpart (Diao and Shen 2010).

### 3 Self-adjusting HSFS

The original HSFS algorithm, despite its competitive performance, relies on a limited set of basic procedures to improvise and search for good quality feature subsets. However, the algorithm can potentially be modified to better support FS. This section details three new components developed to enhance the performance of HSFS.

#### 3.1 Restricted feature domain

In the original HSFS implementation, all musicians  $m_i \in M, i = \{1, \dots, |M|\}$  jointly use a single domain of values,

which is the pool of all features  $A$ . The total number of features  $|A|$  inevitably affects the rate that musicians identify good quality features. The presence of less informative features also reduces the likelihood of locating better features through  $\delta$  activation. As a result, the algorithm may potentially spend unnecessary iterations improvising poor quality candidate solutions, and such emerging feature subsets will be discarded since they introduce no improvement to the *harmony memory*.

A new concept termed the *restricted feature domain* (RFD) is therefore proposed to remedy the aforementioned shortcoming. This mechanism restricts the value domain  $\mathfrak{N}_i$  for any given musician  $m_i$  to a selective subset of  $A$ : the RFDs are constructed during the initialisation phase and reconstructed when the number of musicians is adjusted during the iteration phase. Hence, the recombination of RFDs dynamically affects the choice of musicians throughout the search process. Of course, the union of these RFDs should be equivalent to the full set of features:  $\bigcup_{i \in \{1, 2, \dots, |M|\}} \mathfrak{N}_i = A$  in order to ensure that no important features are mistakenly left out. The feature distribution amongst all the musicians should also be random but uniform. The cardinality of  $\mathfrak{N}_i$  is thus devised to be controlled by a restricted ratio  $\lambda$ ,  $0 < \lambda \leq 1$ , such that  $|\mathfrak{N}_i| = \lceil \lambda \cdot |A| \rceil$ .

A set of RFDs can be generated through various methods, so long as the desired properties that are described above are satisfied. Additionally, if problem domain-specific information is available (e.g., provided by human experts), RFDs may also be populated or adjusted in favour of better quality features. In this paper, for simplicity, an RFD is empirically generated by randomly removing features until  $|\mathfrak{N}_i| = \lceil \lambda \cdot |A| \rceil, \lambda = 0.8$ , while maintaining a full coverage of features across all musicians. The pseudo code of the suggested mechanism is given in Algorithm 1.

#### 3.2 Self-configuration of musician size

The main challenge for FS is to effectively reduce the size of candidate feature subsets, while maintaining their original semantics. The iterative refinement procedure employed by HSFS aims to address this issue (to a certain extent), which is made possible by its flexible mapping of musical concepts onto their associated elements in FS. In particular, a musician is not tied to a specific feature, thereby becoming an independent, single-feature selector. This forms a sharp contrast to many alternative methods that rely on binary-valued feature subset representation.

The iterative refinement procedure first initialises the number of musicians to be the size of the complete set of original features  $|M| = |A|$ . This ensures that no human assumption is involved in the configuration of  $|M|$ , and that the feature subset size may be adjusted according to the actual amount of redundancy present in the data.  $|M|$  is then itera-

**Algorithm 1:** HSFS with RFD

---

$A$ : full set of features  
 $M$ : group of musician  $m_i \in M, i = 1, \dots, |M|$   
 $\delta$ : harmony search considering rate  
 $H_{new}$ : new harmony being improvised  
 $S_i = \bigcup_{j=1}^{\mathbb{H}} H_i^j$ : note domain of musician  $m_i$   
 $\mathbb{S}_i \leftarrow A$ : RFD of musician  $m_i$   
**for**  $i \leftarrow 1$  **to**  $|M|$  **do**  
     **while**  $|\mathbb{S}_i| > \lceil \lambda \cdot |A| \rceil$  **do**  
          $\mathbb{S}_i \leftarrow \mathbb{S}_i \setminus \text{random}(\mathbb{S}_i)$   
     **while**  $\bigcup_{i \in \{1, \dots, |M|\}} \mathbb{S}_i \neq A$  **do**  
         **for**  $i \leftarrow 1$  **to**  $|M|$  **do**  
              $\mathbb{S}_i \leftarrow \mathbb{S}_i \cup \text{random}(A \setminus \bigcup_{i \in \{1, \dots, |M|\}} \mathbb{S}_i)$   
              $\mathbb{S}_i \leftarrow \mathbb{S}_i \setminus \text{random}(\mathbb{S}_i)$   
      $H_{new} \leftarrow \emptyset$ ;  
     **for**  $i \leftarrow 1$  **to**  $|M|$  **do**  
         **if**  $\text{random}(1) < \delta$  **then**  
              $H_{new} \leftarrow H_{new} \cup \text{random}(S_i)$   
         **else**  
              $H_{new} \leftarrow H_{new} \cup \text{random}(\mathbb{S}_i)$   
     **return**  $H_{new}$

---

**Algorithm 2:** Process of HMC

---

**if**  $|M| \neq \max_{j \in \{1, \dots, |\mathbb{H}|\}} (|B_{H^j}|)$  **then**  
      $|M| = \max_{j \in \{1, \dots, |\mathbb{H}|\}} (|B_{H^j}|)$   
     **for**  $H^j \in \mathbb{H}$  **do**  
         **if**  $|M| < |H^j|$  **then**  
             //Consolidation  
             **while**  $|M| < |H^j|$  **do**  
                  $H^j \leftarrow H^j \setminus a_-$   
         **else**  
             //Expansion  
             **while**  $|M| > |H^j|$  **do**  
                  $H^j \leftarrow H^j \cup \{a_-\}$

---

tively reduced (in so doing, the size of feature subsets to be selected is restricted) until no smaller solution can be found without sacrificing the evaluation score. As such, although effective, this procedure leads to repetitive executions of the entire search process, and the earlier executions may also over-restrict the search process to a sub-optimal solution region.

The self-adjusting HSFS algorithm proposed herein embeds an alternative procedure to the aforementioned. It attempts to dynamically and naturally adjust  $|M|$  throughout a single execution of the search, via a means of identifying and eliminating the potentially non-contributing musicians (with their note domains fully filled by duplicated nominations or discarded votes  $a_-$ ). This procedure is referred hereafter as *harmony memory consolidation* (HMC). The pseudocode of HMC is given in Algorithm 2. In particular, to better determine the presence of non-contributing musicians, the following process needs to be performed:

**Table 3** Consolidation of harmony memory

Iteration	Harmony memory						
$k$	$a_1$	$a_-$	$a_2$	$a_-$	$a_1$	$a_2$	$a_3$
	$a_2$	$a_4$	$a_5$	$a_-$	$a_-$	$a_5$	$a_5$
$k$ (HMC)	$a_1$	$a_-$	$a_2$	$a_-$	$a_-$	$a_-$	$a_3$
	$a_2$	$a_4$	$a_5$	$a_-$	$a_-$	$a_-$	$a_7$
$k + 1$	$a_1$	$a_2$	$a_-$	$a_-$	$a_3$		
	$a_2$	$a_4$	$a_5$	$a_-$	$a_7$		

1. The duplicating nominations within each of the harmonies stored in the harmony memory are replaced by  $a_-$ .
2. The desirable value of  $|M|$  may then be derived using the formula given below:

$$|M| = \max_{j \in \{1, \dots, |\mathbb{H}|\}} \{a_i | a_i \in H^j, a_i \neq a_-\} + 1. \quad (1)$$

Alternatively,  $|M|$  may be determined by first converting harmonies  $H^j \in \mathbb{H}$  into feature subsets  $B_{H^j}$ , and then computing:

$$|M| = \max_{j \in \{1, \dots, |\mathbb{H}|\}} (|B_{H^j}|) + 1. \quad (2)$$

3. The existing harmonies are trimmed by randomly removing  $a_- \in H^j$ , until  $|H^j| = |M|$ ,  $H^j \in \mathbb{H}$ .
4. The normal HSFS improvisation process is then resumed, on the basis of the newly consolidated harmony memory.

An illustrative example is given in Table 3. The initial harmony memory (at iteration  $k$ ) consists of several duplicate and discarded nominations, which are identified during the HMC process. For instance, feature subset  $(a_1, a_-, a_2, a_-, a_1, a_2, a_3)$  may be changed into  $(a_1, a_-, a_2, a_-, a_-, a_-, a_3)$  in this given example. The number of musicians is then reduced to  $|M| = \max(3, 4) + 1 = 5$ , and the respective harmonies are also trimmed. The resultant harmony memory after consolidation is then used for the next iteration ( $k + 1$ ).

Note that the HMC procedure may also be utilised to facilitate the expansion of the musician group, because the value of  $|M|$  is determined with respect to the size of the largest feature subset in  $\mathbb{H}$  as shown in Eq. 2. The rationale behind such a mechanism is to allow larger feature subsets to be nominated, if they may lead to a potentially higher (overall) quality solution. Table 4 details an example of harmony memory expansion, where the harmony memory at iteration  $k$  contains a candidate feature subset of size  $|B| = |M|$ . It is probable that there exists more informative feature subsets of size  $|B'| > |M|$ , and therefore, the size of the musician group is enlarged by inserting  $a_-$  at random positions for all  $H \in \mathbb{H}$ .



**Table 4** Expansion of harmony memory

Iteration	Harmony memory				
$k$	$a_1$	$a_2$	$a_3$	$a_-$	
	$a_2$	$a_7$	$a_4$	$a_5$	
$k + 1$	$a_1$	<b><math>a_-</math></b>	$a_3$	$a_-$	$a_2$
	<b><math>a_-</math></b>	$a_7$	$a_4$	$a_5$	$a_2$

Bold texts indicate the randomly inserted discarded notes ( $a_-$ )

### 3.3 Feature subset adjustment using feature similarity measures

The base HS algorithm involves another parameter, termed the pitch adjustment rate (PAR). It offers a stochastic mechanism that allows a note chosen by a given musician to be shifted to a neighbouring one. The underlying motivation of this mechanism is that minor adjustments into neighbouring values may help discover better quality solutions, which is generally true for real-valued optimisation problems. PAR, in conjunction with  $\delta$ , ensures that fine adjustments can be made to an emerging solution, and the solution region may be sufficiently explored.

The original HSFS algorithm does not exploit the benefits offered by PAR. This is because that the values now represent feature indices, each feature and its “neighbours” may not have such general relation, and thus, an adjustment will result in a change into a possibly unrelated feature nearby. However, the absence of PAR hinders the strength of the algorithm in terms of its effectiveness for finding good quality feature subsets. Having recognised this, the pitch adjustment mechanism is re-introduced in this improved HSFS approach together with a method to determine “neighbouring” features. In the context of FS, the use of the PAR parameter  $\omega$ ,  $0 \leq \omega \leq 1$ , will enable a musician to choose a neighbouring feature. Such a feature bears a similarity with the original similarity within the range calculated on the basis of the formula  $(1 - \tau) + \text{Rand}(2 \times \tau)$ , where  $\tau$  is the fret width that constrains the maximal amount of dissimilarity allowed. Note that  $(1 - \omega)$  denotes the probability of using the chosen value without further alteration. Also, as suggested in the original HS algorithm the pitch adjustment procedure and  $\delta$  activation are set to two mutually exclusive events so that further adjustment is only carried out when a feature is selected from within the harmony memory.

To determine the neighbouring features, a number of existing feature similarity measurements may be employed. For example, the non-parameter test-based approaches such as the Walds-Wolfowitz test (Johnson 1966) may use to detect the closeness of probability distributions of the variables. However, such measures are sensitive to both location and dispersion of the distributions (Bhattacharya and Patrange-naru 2002; Maronna et al. 2006), and hence, may not be

suitable to measure the similarity of features in arbitrary data sets. Alternatively, the dependency between the features may be utilised to calculate the degree of feature similarity. In this paper, two existing dependency measures, both linear [*correlation coefficient-based* (Mitra et al. 2002)] and non-linear [*fuzzy-rough set-based* (Jensen and Shen 2009)] are used to obtain the degree of similarities amongst any paired features.

#### 3.3.1 Correlation coefficient-based feature similarity

*Correlation coefficient* is a common linear method for measuring the degree of similarity between two random variables. Correlation coefficient  $\rho$  between two random variables  $x$  and  $y$  is formulated as:

$$\rho(x, y) = \frac{\sigma_{xy}}{\sigma_x \sigma_y} \quad (3)$$

$$\sigma_{xy} = \frac{1}{n-1} \left( \sum_{i=1}^n x_i y_i - n \bar{x} \bar{y} \right) \quad (4)$$

where  $\sigma_x$  and  $\sigma_y$  signify the variance of a variable and  $\sigma_{xy}$  the covariance between two variables. If  $x$  and  $y$  are entirely correlated, then purely linear relationship exists and  $\rho(x, y) \pm 1$ . Independence of  $x$  and  $y$  implies  $\rho(x, y) = 0$ . Hence, the quantity can be used as a measure of similarity between two features (Hsu and Hsieh 2010) with the following properties:

1.  $|\rho(x, y)| \leq 1$ .
2.  $\rho(x, y) = 0$  if and only if  $x$  and  $y$  are linearly correlated.
3.  $\rho(x, y) = \rho(y, x)$ .
4. If  $x^* = ax + b$  and  $y^* = cy + d$  for certain constants  $a, b, c, d$ , then  $\rho(x^*, y^*) = \rho(x, y)$ , implying that the similarity measure is not affected by rescaling and transformation of variables.

The correlation coefficient may be sufficient to the strength of feature similarity, but it makes a strong assumption on linear and highly dependent relationships between features. Non-linear patterns are neglected. For example, assume there exists a quadratic relationship between the value of  $x$  and  $y$  ( $x = y^2$ ), and  $y$  is evenly distributed in the range of  $[-1, 1]$ . The resulting correlation coefficient  $\rho(x, y) = 0$ , indicating that  $x$  and  $y$  are independent. However, there in fact exists a strong dependency between them. Thus, an alternative approach may be necessary.

#### 3.3.2 Fuzzy rough set-based feature similarity

Fuzzy-rough set theory (De et al. 2007; Radzikowska and Kerre 2002) is an extension of traditional rough set theory (Pawlak 1997), where two sets: the lower and upper approximation are defined using fuzzy notions (Dubois and Prade

1992). Rough set is centred upon crisp information granulation. In the crisp case, elements either belong to the lower approximation with absolute certainty or not at all. In the fuzzy-rough case, elements may have a membership in the range  $[0, 1]$ , allowing greater flexibility in handling uncertainty. The concepts of fuzzy lower and upper approximation are defined as follows:

$$\mu_{R_P} X(x) = \inf_{y \in \mathbb{U}} I(\mu_{R_P}(x, y), \mu_X(y)) \quad (5)$$

$$\mu_{\overline{R_P}} X(x) = \sup_{y \in \mathbb{U}} T(\mu_{R_P}(x, y), \mu_X(y)) \quad (6)$$

where  $I$  is a fuzzy impicator and  $T$  a t-norm.  $R_P$  is the fuzzy similarity relation induced by the subset of features  $P$ :

$$\mu_{R_P}(x, y) = \bigcap_{a \in P} \mu_{R_a}(x, y) \quad (7)$$

with  $\mu_{R_a}(x, y)$  being the weight of similarity between instance  $x$  and  $y$  for feature  $a$ . The following three common fuzzy similarity models can be constructed for this purpose.

$$\mu_{R_a}(x, y) = 1 - \frac{|a(x) - a(y)|}{|a_{\max} - a_{\min}|} \quad (8)$$

$$\mu_{R_a}(x, y) = \exp\left(-\frac{(a(x) - a(y))^2}{2\sigma_a^2}\right) \quad (9)$$

$$\mu_{R_a}(x, y) = \max\left(\min\left(\frac{(a(y) - a(x) + \sigma_a)}{\sigma_a}, \frac{(a(x) - a(y) + \sigma_a)}{\sigma_a}\right), 0\right) \quad (10)$$

where  $\sigma_a$  is the standard deviation of feature  $a$ ,  $a_{\max}$  is the maximum value under the feature  $a$ , whereas  $a_{\min}$  is the minimum value. The fuzzy positive region is defined by:

$$\mu_{\text{POS}_{R_P}(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \mu_{R_P} X(x) \quad (11)$$

The resulting degree that a set of features  $Q$  depends on another set of features  $P$  is denoted as  $P \Rightarrow Q$  and is defined as:

$$\gamma_P(Q) = \frac{\sum_{x \in \mathbb{U}} \mu_{\text{POS}_{R_P}(Q)}(x)}{|\mathbb{U}|} \quad (12)$$

Since this paper only concerns similarity measurements between two arbitrary single features, say,  $a_i$  and  $a_j \in A$ . Eq. 12 may be re-written as:

$$\gamma(a_i, a_j) = \frac{\sum_{x \in \mathbb{U}} \mu_{\text{POS}_{R_{\{a_i\}}(\{a_j\})}}(x)}{|\mathbb{U}|} \quad (13)$$

representing the dependency degree of feature  $a_i$  upon  $a_j$ . This formula is used to locate the suitable closest and most

### Algorithm 3: Improvisation Process using PAR

---

$H_{\text{new}}$ : target harmony being improvised  
 $S_i = \bigcup_{j=1}^{\mathbb{H}} H_i^j$ : note domain of musician  $m_i$   
 $\mathbb{S}_i$ : RFD of musician  $m_i$   
 $\gamma(a_p, a_q)$ , feature similarity measure  
**for**  $i \leftarrow 1$  **to**  $|M|$  **do**  
    **if**  $\text{random}[0, 1] < \delta$  **then**  
         $a_p = \text{random}(S_k)$   
        **if**  $\text{random}[0, 1] < \omega$  **then**  
             $\Delta \leftarrow 1 - \tau + \text{random}(2\tau)$   
             $a_q = \underset{a_q \in A, a_q \neq a_p}{\text{argmin}} (\gamma(a_p, a_q) - \Delta)$   
             $H_{\text{new}} \leftarrow H_{\text{new}} \cup \{a_q\}$   
        **else**  
             $H_{\text{new}} \leftarrow H_{\text{new}} \cup \{a_p\}$   
    **else**  
         $H_{\text{new}} \leftarrow H_{\text{new}} \cup \{\text{random}(\mathbb{S}_i)\}$   
**return**  $H_{\text{new}}$

---

dependent neighbouring feature. To ease the understanding, Algorithm 3 shows the pseudo-code of the improvisation process using pitch adjustment.

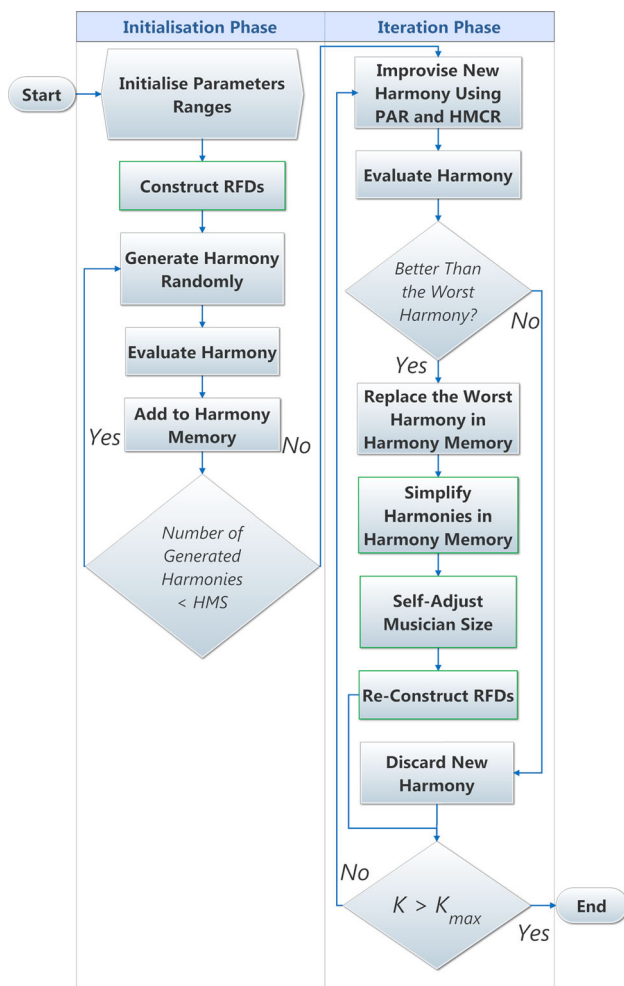
### 3.4 Self-adjusting HSFS

The procedure of the complete self-adjusting HSFS algorithm is illustrated in Fig. 2. It incorporates three new modules: RFD construction, harmony memory consolidation, and PAR-based feature subset improvisation. The complexity of the HSFS algorithm is analysed below. The initialisation requires  $O(|M| \times |\mathbb{H}|)$  operations to randomly populate the subset storage, and the improvisation process is of the order  $O(|M| \times k_{\max})$  because every feature selector needs to produce a new feature at every iteration. Here,  $|\mathbb{H}|$  is the subset storage size,  $|M|$  is the number of feature selectors, and  $k_{\max}$  is the maximum number of iterations. The construction of restricted feature domains requires  $O(|M| \times k_{\max} \times [\lambda \times |A|])$ , which occurs in both initialisation phase and iteration phase when the number of musicians is changed. The HMC procedure has a computational complexity of  $O(|\mathbb{H}| \times |M|)$  since all stored harmonies need to be examined and consolidated. The additional overhead introduced by the PAR process is largely due to the feature similarity matrix construction, which incurs a cost of up to  $O(|A|^2 \times |X|^2)$ , where  $|X|$  is the total number of training the instances in data set. Thus, the overall algorithm complexity (including the initial cost of computing the feature similarity values) is:

$$O(|A|^2 \times |X|^2) + O(|M| \times (|\mathbb{H}| + [\lambda |A|]) \times k_{\max}) \quad (14)$$

## 4 Experimentation and discussion

In this section, the results of experimental investigations are reported to demonstrate the capabilities and characteristics



**Fig. 2** Proposed self-adjusting HSFS algorithm

of the proposed improvements. Notationally, in the results, HSFS<sub>SA</sub> stands for the whole algorithm that incorporates all of the newly proposed mechanisms. Section 4.1 presents a comparison against other search methods, including the original HSFS (HSFS<sub>O</sub>) and two nature-inspired optimisation techniques: genetic algorithms (GAs) (Leardi et al. 1992) and particle swarm optimisation (Chuang et al. 2008). Two filter-based feature subset evaluators with score values  $f(B)$ ,  $0 \leq f(B) \leq 1$ : the correlation-based (CFS), and the probabilistic consistency-based (PCFS) are employed. The experimentation uses a total of 10 real-valued UCI benchmark data sets (Frank and Asuncion 2010), several of which are of high dimensionality and/or contain a large number of objects, thereby presenting reasonably realistic challenges for the proposed techniques. A summary of these data sets is provided in Table 5. Parameter setting in the investigated algorithms are presented in Table 6. Finally, the C4.5 algorithm (Witten and Frank 2005) is adopted due to its popularity to verify the quality of the selected feature subsets from an

**Table 5** Data set properties

Data set	Features	Objects	Decisions
Arrhythmia	280	452	13
Handwritten	257	1593	10
Ionosphere	35	230	2
Libras	91	360	15
Multifeat	650	2000	10
Secom	591	1567	2
Segment	20	1500	7
Sonar	61	208	2
Cnae	857	1080	9
Web	2557	149	5

**Table 6** Parameter configurations

HSFS <sub>SA</sub>	$ \mathbb{H} $	15
	$\lambda$	0.8
	$\delta$	0.8
	$k_{\max}$	5000
HSFS <sub>O</sub>	$ \mathbb{H} $	15
	$ \mathbb{M} $	$ \mathbb{A} $
	$\delta$	0.8
	$k_{\max}$	5000
GA	Cross over	0.6
	Populations	20
	Mutation	0.033
	Max generation	5000
PSO	Particles	100
	C1	1
	C2	2
	Max generation	5000

end classifier learner's perspective, where stratified ten-fold cross-validation is exploited for accuracy validation.

#### 4.1 Comparison with alternative search strategies

Stratified tenfold cross-validation (10-FCV) is used. For a given data set, it works by dividing the data into ten sub-tables. Nine of these ten sub-folds are employed for training, where FS is employed to learn the optimal feature subset. The remaining single fold is used to test the classifier using the selected feature subset. This process is then repeated ten times. Therefore, each fold is used for testing only once and the stratification of the data ensures that each class label has the same representation in all folds, thereby helping to alleviate bias/variance problems (Bengio and Grandvalet 2004). In the experiments, 10-FCV is performed using ten different random folds of the data in order to lessen the impact of ran-



dom factors within the heuristic algorithms. These  $10 \times 10$  sets of evaluations are aggregated to produce the final experimental outcomes.

In addition, a paired  $t$  test with two-tailed  $p = 0.01$  has been performed in order to compare the statistical differences between results obtained by HSFS<sub>O</sub> and HSFS<sub>SA</sub>, as given in Table 7. The symbol ‘ $v$ ’ indicates that HSFS<sub>SA</sub> obtains a better results than HSFS<sub>O</sub>, ‘ $-$ ’ denotes that there exists no statistical difference between the results, and ‘ $*$ ’ signifies that HSFS<sub>SA</sub> results in statistically worse search performance. These comparisons are made in terms of whether the generated feature subsets offer higher evaluation score, smaller subset size and/or better classification accuracy.

As reflected in Table 7, for CFS, higher evaluation scores are obtained using the proposed techniques for six of the ten data sets (indicated with  $v$ ) when compared to HSFS<sub>O</sub>. For the remaining four data sets: *ionosphere*, *secom*, *segment*, and *sonar*, more compact subsets are discovered with equal evaluation scores (or with the tiny difference in evaluation score). The bold figures signify improved performance when compared with HC, where HSFS<sub>SA</sub> enables further reduction of the size of the selected feature subsets. Although the GAs are capable of identifying higher quality subsets for several data sets, they are unable to identify good quality solutions for the higher dimensional data sets such as *arrhythmia*, *multifeat*, *secom*, *cnac*, and *web*. Importantly, the classification accuracy of the classifier learners built using the reduced feature subsets (selected by HSFS<sub>SA</sub>) is improved, particularly for *arrhythmia* (by 13.40 %), *sonar* (by 5.04 %), and *web* (by 5.43 %).

Note that the execution time of HSFS<sub>SA</sub> is longer than HSFS<sub>O</sub> for most of the data sets except *sonar* and *web*, which is caused by the complexities incurred because of the introduced self-adjusting components. This observation confirms the complexity analysis performed in Sect. 3.4.

For the results obtained using the PCFS evaluator, the performance improvement over the original algorithm is more obvious, where better subsets are selected across all data sets. Note that only a single feature is selected by HC for the *secom* data set. This is because no additional features offer any increase in the evaluation score, when combined with this selected feature. However, better combinations of features do exist, which are successfully identified by all other employed stochastic approaches. Feature subsets obtained by GAs are mostly comparable to the others in terms of the evaluation scores. GAs are indeed able to identify feature subsets of optimal quality for data set *libras*. The classifiers built using the feature subsets selected by HSFS<sub>SA</sub> also show improved classification performance for six of the ten data sets. Interestingly, higher classification accuracies are obtained by HSFS<sub>SA</sub> for nine of the whole ten data sets when compared to HSFS<sub>O</sub>. For the remaining data set, *libras*, the

minor differences in accuracy are acceptable, given the substantial reduction in the averaged feature subset size (16.40 for HSFS<sub>SA</sub>, and 57.23 for HSFS<sub>O</sub>).

#### 4.2 Effect of individual strategy

Two of the data sets with the largest number of features: *multifeat* and *secom* are employed for the remainder of the experimentation. Each of the proposed modifications to HSFS is tested on its own in order to ascertain what benefits these strategies offer individually. Figure 3 illustrates the effects of the restricted ratio  $\lambda$  which controls the size of the RFDs. The results are collected using different values for  $\lambda$  ranging from 0.1 to 1, with an interval of 0.1. Intuitively, if the value of  $\lambda$  is too small, the musicians may have too few features to work with, which will limit the solution quality. For both data sets, the quality of the selected subsets approach peaks at  $\lambda = 0.8$ , when 80 % of features in  $A$  are utilised.

Figure 4 aims to demonstrate the effectiveness of the HMC process. The algorithm successfully adjusts the number of musicians to a reasonable level, down from the initial setting of  $|M| = |A|$ , without subjective intervention. The reduced group size further encourages the remaining feature selectors to identify even smaller candidate solutions. The compactness of the resulting subsets as reported in Table 7 also provides good evidence for the positive impacts of this process.

The final set of experiments is carried out in order to study the effects of the pitch adjustment rate, by varying  $\omega$  from 0 to 1 with an interval of 0.1. This parameter manipulates the shift ratio, where a feature is replaced with a random close neighbouring feature. Figure 5 shows the size and evaluation scores of the best solutions recorded during the search process. The performance of both data sets: *multifeat* and *secom* peaks at  $\omega = 0.2$ , when a feature shifts to neighbouring features with a probability of 20 %. Obviously, if the value of  $\omega$  is set too large, good quality features may be neglected.

## 5 Conclusion

This paper has presented a self-adjusting FS search algorithm which improves the original harmony search-based feature selection (Diao and Shen 2012) with three new techniques. The proposed techniques are conceptually simple and require limited computational overheads in order to achieve positive effects. The musicians in HSFS<sub>SA</sub> improvise new candidate feature subsets using a selective portion of the full set of original features (RFD), and dynamically adjust their subset size via the HMC process. The pitch adjustment strategy, re-introduced to HSFS via feature similarity measures allows finer yet relevant adjustments to emerging feature subsets. Experimental results show that the added enhancements can

**Table 7** Comparison using CFS and PCFS ( $10 \times 10$ ) cross-validation, with respect to average subset size, evaluation score, classification accuracy (%), and execution time (millisecond)

CFS										
Dataset	Full	HSFSSA			HSFS <sub>0</sub>			C4.5%	Time	
		A	C4.5%	Size	Eval.	C4.5%	Time			
Arrhythmia	280	64.38		<b>12.0</b>	<b>0.449</b> (v)	77.78 (v)	1792	22.9	62.89	1138
Handwritten	257	75.83		<b>29.3</b>	<b>0.524</b> (v)	76.39 (v)	1650	63.8	75.45	954
Ionosphere	35	87.83		<b>8.7</b> (v)	<b>0.536</b>	89.56 (v)	46	11.1	88.43	37
Libras	91	69.72		<b>15.4</b>	<b>0.613</b> (v)	68.00 (*)	817	27.3	68.04	152
Multifeat	650	94.75		<b>83.1</b>	0.920 (v)	95.12 (v)	9413	81.6	93.50	5587
Secom	591	89.60		<b>15.2</b> (v)	<b>0.096</b>	90.44 (v)	6754	36.8	90.11	4557
Segment	20	95.73		<b>4.5</b> (v)	<b>0.734</b>	94.80 (—)	26	5.1	<b>0.732</b>	17
Sonar	61	71.15		<b>9.4</b> (v)	0.352	76.19 (v)	76	17.9	73.72	80
Cnae	857	88.8		14.4	<b>0.426</b> (v)	77.80 (v)	155363	18.6	69.80	139857
Web	2557	51.6		<b>33.3</b>	0.543 (v)	57.03 (v)	97512	35.4	55.33	135671
GA										
Dataset	Size	Eval.	PSO			HC			Eval.	
			Time	Size	Eval.	Time	Size			
Arrhythmia	46.2	0.397	5418	13.7	0.288	1371	25	0.438		
Handwritten	93.9	<b>0.523</b>	8044	127.3	0.472	3416	75	<b>0.524</b>		
Ionosphere	10.3	<b>0.539</b>	180	10.3	0.535	195	11	0.521		
Libras	30.5	<b>0.611</b>	984	26.5	0.592	435	26	0.608		
Multifeat	217.9	0.894	59173	315.2	0.854	27802	144	<b>0.925</b>		
Secom	51.2	0.012	28847	27.5	0.013	16035	17	<b>0.096</b>		
Segment	<b>4.4</b>	<b>0.732</b>	80	<b>4.4</b>	0.735	132	6	0.725		
Sonar	17.5	<b>0.359</b>	412	11.4	0.327	290	19	0.352		
Cnae	170.9	0.037	65066	<b>10.8</b>	0.073	23613	28	0.414		
Web	96.0	0.327	169760	100.9	0.129	27068	59	<b>0.565</b>		

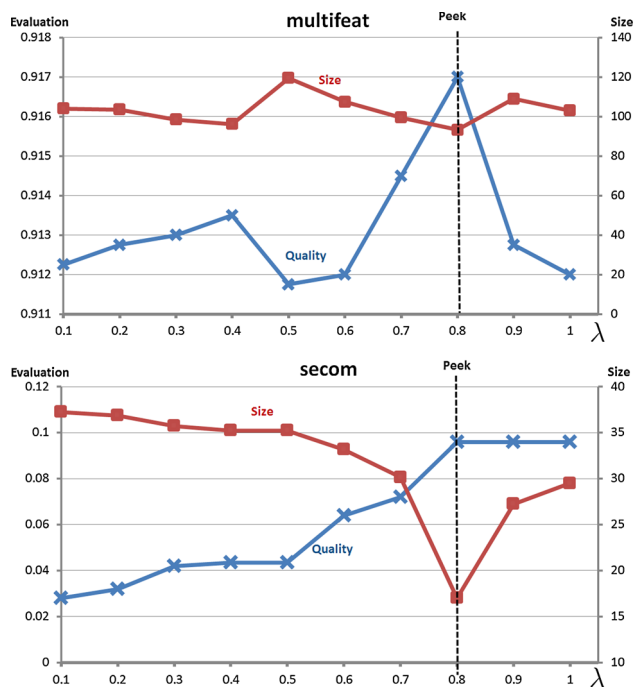
**Table 7** continued

Dataset	Base		HSFS <sub>SA</sub>		HSFS <sub>So</sub>		Time	C4.5%	Eval.	Size	C4.5%	Time
	A											
Arrhythmia	280	64.38	29.1 (v)	<b>0.989</b>	67.59 (v)	4202			<b>0.989</b>	136.2	64.16	4336
Handwritten	257	75.83	70.2 (v)	<b>1.000</b>	79.38 (v)	155			<b>1.000</b>	80.0	75.07	140
Ionosphere	35	87.83	<b>6.8</b> (v)	<b>0.997</b>	87.10 (v)	444			<b>0.997</b>	21.5	86.70	394
Libras	91	69.72	<b>16.4</b>	<b>0.978</b> (v)	68.75 (v)	1198			0.974	57.2	69.33	1234
Multifeat	650	94.75	9.1 (v)	<b>1.000</b>	95.58 (v)	424			<b>1.000</b>	26.2	94.35	407
Secom	591	89.60	23.7 (v)	<b>0.990</b>	89.42 (v)	29084			<b>0.989</b>	154.9	88.70	30858
Segment	20	95.73	<b>6.4</b> (v)	<b>0.998</b>	95.91 (v)	3044			0.997	8.5	95.6	2710
Sonar	61	71.15	<b>11.7</b>	<b>1.000</b> (v)	74.72 (v)	552			0.989	14.5	73.2	534
Cnae	857	88.8	<b>65.2</b> (v)	<b>0.983</b>	85.19 (v)	122361			<b>0.981</b>	69.0	76.85	106798
Web	2557	51.6	<b>18.6</b> (v)	<b>1.000</b>	62.22 (v)	101459			<b>1.000</b>	21.1	54.45	94431
PSO												
Dataset	GA											
	Size	Eval.	Time	Size	Eval.	Time	Size	Eval.	Time	Size	Eval.	Time
Arrhythmia	29.2	<b>0.989</b>	34957	112.5	<b>0.989</b>	4514	<b>21</b>	0.988				
Handwritten	33.0	<b>1.000</b>	127	23.5	<b>1.000</b>	5463	<b>18</b>	<b>1.000</b>				
Ionosphere	10.0	<b>0.997</b>	85	8.9	<b>0.997</b>	334	<b>7</b>	0.996				
Libras	17.3	0.974	11559	28.2	0.973	1099	17	0.969				
multifeat	43.0	<b>1.000</b>	222	12.2	<b>1.000</b>	10535	<b>7</b>	<b>1.000</b>				
Secom	92.2	<b>0.989</b>	291034	284.8	<b>0.990</b>	30571	<b>1</b>	0.936				
Segment	7.0	0.997	2152	7.5	0.997	597	9	<b>0.998</b>				
Sonar	<b>12.0</b>	0.989	5132	18.5	0.989	579	14	0.981				
Cnae	213.9	<b>0.983</b>	324843	699.1	<b>0.983</b>	52925	67	<b>0.981</b>				
Web	1036.8	<b>1.000</b>	778	585.7	<b>1.000</b>	26575	21	<b>1.000</b>				

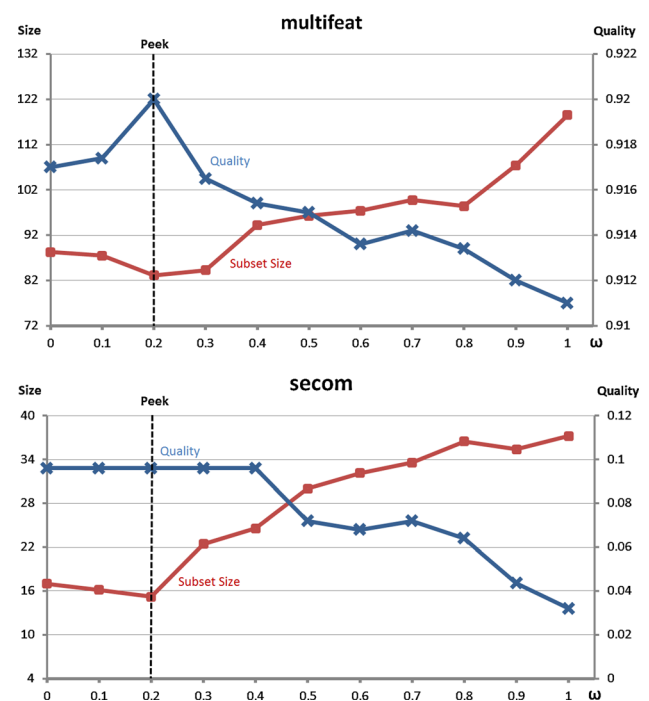
v, -, and \* indicate statistically better, equal, and worse results respectively, when compared to the original algorithm

Bold figures signify the overall best results obtained for each of the data sets

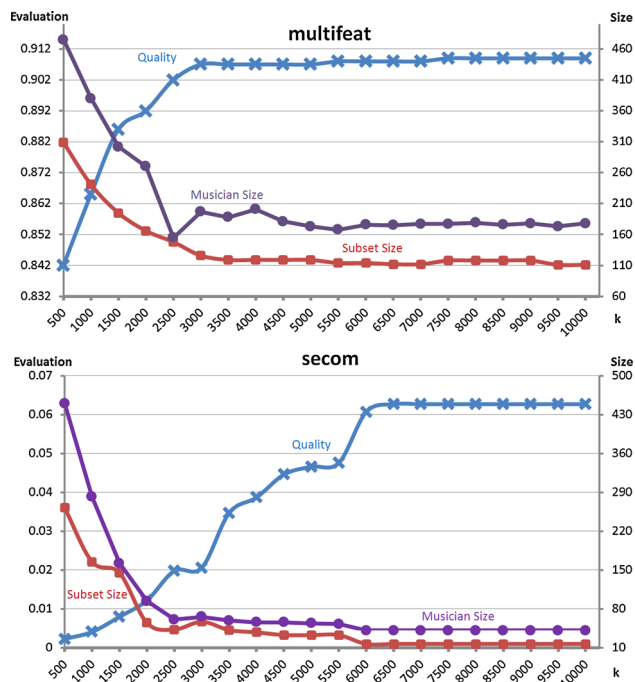
Italic figures indicate that higher classification accuracies have been achieved as a result of feature selection, when compared with the accuracies obtained using the original data set of full features



**Fig. 3** Demonstration of the effects of RFD using different  $\lambda \in [0.1, 1]$



**Fig. 5** Demonstration of the effects of PAR using different  $\omega \in [0, 1]$



**Fig. 4** Automatic configuration of musician size using HMC

indeed further improve the quality of the resultant feature subsets (when compared to the original HSFS approach), and also obviate the need to precisely pre-configure the size of the musician group.

Although promising, much can be done to further refine the proposed improvements. While the possibility of ex-

changing information between musicians has been explored in the HMC procedure, there may exist alternative applications of this mechanism (e.g., promoting high quality features, or preserving minority features). Note that these proposed improvements may also be employed by other nature-inspired FS search algorithms. In particular, the PAR mechanism of HS is conceptually similar to mutation operators used by GAs and PSO. Alternative feature similarity measures are also worth investigating, which may prove to be more efficient than fuzzy-rough set-based measures. Finally, theoretical extensions to the techniques in the area of dynamic FS, classifier ensemble reduction with FS, and feature grouping are also of interest in further developing this work.

**Acknowledgments** The first author is supported by an Aberystwyth University PhD scholarship, and the second by an Aberystwyth University research fellowship. The authors are grateful to the reviewers who provided the constructive comments on the original version of this paper, which have helped significantly in modifying this research.

## References

- Aarts E, van Laarhoven P (1989) Simulated annealing: an introduction. *Stat Neerl* 43(1):31–52
- Al-Betar M, Khader A, Zaman M (2012) University course timetabling using a hybrid harmony search metaheuristic algorithm. *IEEE Trans Syst Man Cybern C* 42(5):664–681
- Bellman R (1957) *Dynamic programming*, 1st edn. Princeton University Press, Princeton
- Bengio Y, Grandvalet Y (2004) No unbiased estimator of the variance of k-fold cross-validation. *Mach Learn Res* 5:1089–1105

- Bhattacharya R, Patranenaru V (2002) Nonparametric estimation of location and dispersion on riemannian manifolds. *J Stat Plan Inference* 108(1):23–35
- Chuang LY, Chang HW, Tu CJ, Yang CH (2008) Improved binary PSO for feature selection using gene expression data. *Comput Biol Chem* 32(1):29–38
- De Cock M, Cornelis C, Kerre EE (2007) Fuzzy rough sets: the forgotten step. *IEEE Trans Fuzzy Syst* 15(1):121–130
- Das S, Mukhopadhyay A, Roy A, Abraham A, Panigrahi B (2011) Exploratory power of the harmony search algorithm: analysis and improvements for global numerical optimization. *IEEE Trans Syst Man Cybern B* 41(1):89–106
- Dash M, Liu H (2003) Consistency-based search in feature selection. *Artif Intell* 151(1–2):155–176
- Diao R, Shen Q (2012) Feature selection with harmony search. *IEEE Trans Syst Man Cybern B* 42(6):1509–1523
- Diao R, Shen Q (2010) Two new approaches to feature selection with harmony search. In: *IEEE interantional conference on fuzzy systems*, pp 1–7
- Dubois D, Prade H (1992) Putting rough sets and fuzzy sets together. *Intelligent Decision Support*, Kluwer Academic, Dordrecht
- Fesanghary M, Mahdavi M, Minary-Jolandan M, Alizadeh Y (2008) Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. *Comput Methods Appl Mech Eng* 197(33–40):3080–3091
- Frank A, Asuncion A (2010) UCI machine learning repository
- Geem ZW (ed) (2010) Recent advances in harmony search algorithm. In: *Studies in computational intelligence*, vol 270. Springer, Berlin
- Grefenstette JJ (1986) Optimization of control parameters for genetic algorithms. *IEEE Trans Syst Man Cybern* 16(1):122–128
- Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. *J Mach Learn Res* 3:1157–1182
- Hall MA (1998) Correlation-based feature subset selection for machine learning. PhD thesis, University of Waikato, Hamilton, New Zealand
- Hsu HH, Hsieh CW (2010) Feature selection via correlation coefficient clustering. *J Softw* 5(12):1371–1377
- Jensen R, Shen Q (2009) Are more features better? A response to attributes reduction using fuzzy rough sets. *IEEE Trans Fuzzy Syst* 17(6):1456–1458
- Jensen R, Shen Q (2009) New approaches to fuzzy-rough feature selection. *IEEE Trans Fuzzy Syst* 17(4):824–838
- Johnson N (1966) Linear statistical inference and its applications. *Technometrics* 8(3):551–553
- Leardi R, Boggia R, Terrile M (1992) Genetic algorithms as a strategy for feature selection. *J Chemom* 6(5):267–281
- Lee KS, Geem ZW (2004) A new structural optimization method based on the harmony search algorithm. *Comput Struct* 82(9):781–798
- Liu H, Motoda H (2008) Computational methods of feature selection. Chapman & Hall/CRC, USA
- Mac Parthaláin N, Jensen R, Shen Q, Zwiggelaar R (2010) Fuzzy-rough approaches for mammographic risk analysis. *Intell Data Anal* 14(2):225–244
- Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. *Appl Math Comput* 188(2):1567–1579
- Mahdavi M, Chehreghani MH, Abolhassani H, Forsati R (2008) Novel meta-heuristic algorithms for clustering web documents. *Appl Math Comput* 201(1–2):441–451
- Maronna RA, Martin RD, Yohai VJ (2006) Robust statistics. Wiley, New York
- Mashinchi M, Orgun MA, Mashinchi M, Pedrycz W (2011) A tabu-harmony search-based approach to fuzzy linear regression. *IEEE Trans Fuzzy Syst* 19(3):432–448
- Mitra P, Murthy C, Pal SK (2002) Unsupervised feature selection using feature similarity. *IEEE Trans Pattern Anal Mach Intell* 24(3):301–312
- Muni DP, Pal NR, Das J (2006) Genetic programming for simultaneous feature selection and classifier design. *IEEE Trans Syst Man Cybern B* 36(1):106–117
- Pawlak Z (1997) Rough set approach to knowledge-based decision support. *Eur J Oper Res* 99(1):48–57
- Radzikowska AM, Kerre EE (2002) A comparative study of fuzzy rough sets. *Fuzzy Sets Syst* 126(2):137–155
- Rao Srinivasa R, Narasimham SVL, Ramalinga Raju M, Srinivasa Rao A (2011) Optimal network reconfiguration of large-scale distribution system using harmony search algorithm. *IEEE Trans Power Syst* 26(3):1080–1088
- Shah M, Marchand M, Corbeil J (2012) Feature selection with conjunctions of decision stumps and learning from microarray data. *IEEE Trans Pattern Anal Mach Intell* 34(1):174–186
- Shang C, Barnes D (2013) Fuzzy-rough feature selection aided support vector machines for mars image classification. *Comput Vision Image Underst* 117(3):202–213
- Sharma Das K, Chatterjee A, Rakshit A (2010) Design of a hybrid stable adaptive fuzzy controller employing lyapunov theory and harmony search algorithm. *IEEE Trans Control Syst Technol* 18(6):1440–1447
- Shen Q, Jensen R (2004) Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring. *Pattern Recognit* 37(7):1351–1363
- Tsang EC, Chen D, Yeung DS, Wang XZ, Lee J (2008) Attributes reduction using fuzzy rough sets. *IEEE Trans Fuzzy Syst* 16(5):1130–1141
- Vasebi A, Fesanghary M, Bathaee SMT (2007) Combined heat and power economic dispatch by harmony search algorithm. *Int J Electr Power Energy Syst* 29(10):713–719
- Wang X, Yang J, Teng X, Xia W, Jensen R (2007) Feature selection based on rough sets and particle swarm optimization. *Pattern Recognit Lett* 28(4):459–471
- Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn., Morgan Kaufmann series in data management systems Morgan Kaufmann, San Francisco
- Xing EP, Jordan MI, Karp RM (2001) Feature selection for high-dimensional genomic microarray data. In: *Proceedings of the eighteenth international conference on machine learning*, pp 601–608. Morgan Kaufmann, San Francisco
- Yang, XS (2009) Harmony search as a metaheuristic algorithm. In: *Music-inspired harmony search algorithm*, pp 1–14. Springer, Berlin
- Zhang R, Hanzo L (2009) Iterative multiuser detection and channel decoding for ds-cdma using harmony search. *IEEE Signal Process Lett* 16(10):917–920
- Zheng L, Diao R, Shen Q (2013) Efficient feature selection using a self-adjusting harmony search algorithm. In: *Proceeding of the 2013 13th UK workshop on computational intelligence*, pp 167–174