Metadata of the chapter that will be visualized in SpringerLink

Book Title	Intelligent Systems and Applications	
Series Title		
Chapter Title	An Exploratory Study of the Inputs for Ensemble Clustering Technique as a Subset Selection Problem	
Copyright Year	2019	
Copyright HolderName	Springer Nature Switzerland AG	
Corresponding Author	Family Name	**Ayed**
	Particle	
	Given Name	**Samy**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Brunel University London
	Address	Middlesex, UK
	Email	samy.ayed@brunel.ac.uk
Author	Family Name	**Arzoky**
	Particle	
	Given Name	**Mahir**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Brunel University London
	Address	Middlesex, UK
	Email	
Author	Family Name	**Swift**
	Particle	
	Given Name	**Stephen**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Brunel University London
	Address	Middlesex, UK
	Email	
Author	Family Name	**Counsell**
	Particle	
	Given Name	**Steve**
	Prefix	
	Suffix	

	Role	
	Division	
	Organization	Brunel University London
	Address	Middlesex, UK
	Email	
Author	Family Name	**Tucker**
	Particle	
	Given Name	**Allan**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Brunel University London
	Address	Middlesex, UK
	Email	

Abstract	Ensemble and Consensus Clustering address the problem of unifying multiple clustering results into a single output to best reflect the agreement of input methods. They can be used to obtain more stable and robust clustering results in comparison with a single clustering approach. In this study, we propose a novel subset selection method that looks at controlling the number of clustering inputs and datasets in an efficient way. The authors propose a number of manual selection and heuristic search techniques to perform the selection. Our investigation and experiments demonstrate very promising results. Using these techniques can ensure better selection methods and datasets for Ensemble and Consensus Clustering and thus more efficient clustering results.

Keywords (separated by '-')	Ensemble clustering - Consensus clustering - Subset selection problem - Heuristic search - Machine learning

An Exploratory Study of the Inputs for Ensemble Clustering Technique as a Subset Selection Problem

Samy Ayed(✉), Mahir Arzoky, Stephen Swift, Steve Counsell, and Allan Tucker

Brunel University London, Middlesex, UK
samy.ayed@brunel.ac.uk

AQ1

Abstract. Ensemble and Consensus Clustering address the problem of unifying multiple clustering results into a single output to best reflect the agreement of input methods. They can be used to obtain more stable and robust clustering results in comparison with a single clustering approach. In this study, we propose a novel subset selection method that looks at controlling the number of clustering inputs and datasets in an efficient way. The authors propose a number of manual selection and heuristic search techniques to perform the selection. Our investigation and experiments demonstrate very promising results. Using these techniques can ensure better selection methods and datasets for Ensemble and Consensus Clustering and thus more efficient clustering results.

Keywords: Ensemble clustering · Consensus clustering
Subset selection problem · Heuristic search · Machine learning

1 Introduction

Clustering is the process of differentiating groups inside a given set of objects. The resulting groups are assigned so that objects that are with each subset are more closely related to each other than objects that are assigned in different subsets. There are various practical applications involving the grouping of a set of objects into a number of non-overlapping subsets. Partitioning methods formulated on the relationship between objects through correlation or other distance metrics are mutually known as clustering algorithms [18]. There is extensive work in the field of clustering, with many clustering algorithms being developed. Each one of these algorithms can utilise different similarity methods and/or have a different objective function. In addition, varying the parameters of the same method or different methods and applying it on the same data can produce varying results. Moreover, clustering methods can perform well on some datasets but not on others. Thus, an essential question to ask is: given the number of clustering methods and datasets, how do we choose between them?

One way to solve the varying clustering results is through the use of input formulated from multiple clustering results, a technique called Ensemble Clustering that has been gaining popularity recently. The Ensemble Clustering problem aims to operate summation in order to gain a representative clustering with the least variability and an augmentation in mutual agreement. Ensemble Clustering overcomes inherent biases in the search method by relaxing constraints to accept a solution rated as poorer than

K. Arai et al. (Eds.): IntelliSys 2018, AISC 868, pp. 1–15, 2019.
https://doi.org/10.1007/978-3-030-01054-6_72

neighbouring solutions. This allows clustering algorithms to expand their local maxima and further explore the search space beyond local bounds. The ensemble methodology explores beyond the local maxima through the use of an input, formulated from multiple clustering results to return a clustering result based upon the agreement between these inputs.

Ensemble Clustering was first introduced by Strehl and Ghosh [27], has gained great momentum and has been covered in the literature of [1, 5, 7, 12, 14, 15, 23]. This problem relates to a structure which targets a combination of a set of multiple clustering solutions or partitions into a single concrete clustering which optimises the data shared amongst all the accessible clustering solutions. Undoubtedly, single clustering may not always generate promising results and similarly, multiple algorithms may independently make inferior choices by assigning some elements to the wrong clusters [25]. However, by taking into account the outcome of several different clustering algorithms jointly, better results can be achieved by mitigating degeneracies in distinctive solutions. These problems have been further investigated in the literature of [10, 13, 21] in the context of the stability and accuracy of the results. The Ensemble Clustering technique has been shown to work on different datasets and especially well in bioinformatics [28]. It has highlighted the potential of certain algorithms in applications to synthetic datasets. While ensemble clustering is becoming ubiquitous in other aspects of data mining, however, there has been little research into the implementation of an ensemble approach in the area of cluster analysis [28].

Consensus Clustering (CC) is an ensemble method which uses an agreement matrix to serve as a communal reservoir of clustering inputs, from which a consensus can be iteratively obtained. CC receives a number (r), of clustering results as input and returns a single consensus based on the level of agreement between these inputs. It is oriented towards generating an optimum solution by iteratively comparing pairwise clustering methods to determine a level of agreement. CC subsequently filters out poor agreements until an optimised clustering ensemble is generated.

For this project, we look to model the behaviour of a variety of clustering methods (inputs for consensus clustering) and datasets to create a large number of synthetic datasets for investigating consensus clustering and the way it performs with the inputs. However, before this can be accomplished it is difficult to get representative datasets just by performing experiments on all datasets since they are of different properties and sizes. In addition, datasets can work well on some clustering techniques and not others.

A number of studies have looked at the cluster ensemble selection problem [4, 6, 11]. Given a large library of clustering methods, the problem looks at identifying and selecting a subset from the library to produce a smaller cluster ensemble with an equivalent or better clustering methods performance. The majority of studies use metrics that may influence the performance of ensemble clustering such as diversity, quality or accuracy to design the methods. However, there seems to be a lack of approaches which look at identifying the optimal subset of both clustering inputs (for Ensemble Clustering) and datasets. We propose a different metric, Weighted Kappa (WK), which measures agreements between the clustering inputs for all selected datasets [2]. One advantage of using the WK metric is that there is an interpretable table that can provide input on the quality of the results. To our knowledge, this is the first study that makes use of this

metric in the cluster ensemble selection problem, in addition, no previous study has looked at selecting both the clustering inputs and datasets for CC using heuristic search techniques. Hence, this paper investigates a novel combinatorial optimisation technique that looks at identifying and selecting a suitable subset for benchmarking and testing Ensemble clustering by controlling the number of inputs and datasets in a much more efficient manner. We propose two manual selection methods and three heuristic techniques (Genetic Algorithm, Simulated Annealing and Hill Climbing) to perform the selection. We believe that by using heuristic algorithms it is possible to obtain better selection results. A quality metric needs to be defined to maximise the number of inputs and datasets to include in CC. We look to investigate if the inputs and datasets chosen are enough to produce a model out of them i.e., evaluate the results to see if there is a distribution that they fit into. We look to extend this work to explore and assess the CC methodology for evaluating the efficiency of a given clustering method clusters more comprehensively.

The paper is organised as follows. In the next section, we describe the datasets and the clustering techniques chosen for this study. In Sect. 3, we introduce our quality measurement metric, Weighted Kappa. Section 4 details the experimentations conducted to select the appropriate WK threshold. The process of matrix creation and data preparation is introduced in Sect. 5. Section 6 details the experimental methods chosen for this study. A total of five proposed methods are introduced in this section. Section 7 explains the results produced from the experiments and Sect. 8 presents the post-analysis results. Finally, Sect. 9 gives a brief description of our conclusions and presents future directions.

2 Datasets and Consensus Clustering Inputs

2.1 Datasets

The datasets used for this paper are derived from various data repositories used by the machine learning community for the empirical analysis of machine learning algorithms. Particular attention is given to the type and nature of the datasets selected with strong emphasis on real-world data. We collated a wide range of data categories mainly clustering data that includes; bio-medical, statistical, botanical and ecological. Data were collected from: MLData [22], UCI Machine Learning Repository [9], Kaggle Datasets [19], StatLib [26] and Time Series Data Library [17].

The database currently contains 198 datasets, with attributes (number of columns) ranging from 3 to 167 and instances (number of rows) up to 4,898. All datasets went through a data cleansing process to make sure that they were accurate and in the correct format before running them on the clustering methods. The authors would like to note that the expected clustering arrangements are known for all datasets.

2.2 Clustering Methods (Inputs)

There are broad classes of traditional approaches to the clustering problem, for this work we present a wide range of different clustering algorithms (input methods), 32 in total.

This allowed us to be more confident about the reliability of our methods. R was used to implement the inputs for CC, which in this case was used on our subset selection problem. The R script produces the 32 inputs selected and the expected clustering arrangement for the 198 datasets. Table 1 displays a summary of the input methods selected for this work and the number of variations implemented for each method.

Table 1. Clustering methods (inputs) summary

Clustering methods	Details	Variations
K-means	The 'stats' package is used for implementing the K-means function. The following algorithms were used: Forgy, Lloyd, MacQueen and Hartigan-Wong	4
Hierarchical clustering	The agglomeration methods are Ward, Single, Complete, Average, Mcquitty, Median and Centroid. Two versions of the methods are produced, using both Euclidian and Correlation distance methods. The 'stats' package is used	14
Model-based clustering	Model-based clustering is implemented using a contributed R package called 'mclust'. The following identifiers is used VII, EEI, VVI, EEV and VVV	5
Affinity Propagation (AP)	An R package for AP clustering called 'apcluster' is used. AP was computed using the following similarity methods: negDistMat, expSimMat and linSimMat	3
Partitioning Around Medoids (PAM)	A more generic version of the K-means method is implemented using the 'cluster' package. Two similarity distance methods are used: Euclidean and Correlation	2
Clara (partitioning clustering)	Clara is a partitioning clustering method for large applications. It is part of the 'cluster' package	1
X-means clustering	An R Script based on [24]	1
Density Based Clustering of Applications with Noise (DBSCAN)	A density-based algorithm as part of the 'dbscan' package	1
Louvain clustering	A multi-level optimisation of modularity algorithm for finding community structure	1

3 Weighted-Kappa

WK is a simple statistical metric derived from Cohen's Kappa Coefficient of Agreement [8]. It is used for measuring the inter-rater agreement between two or more observers. In clustering problems, this allows for a comparative assessment of two or more components. Moreover, the class relationships between clustering arrangements generated through multiple clustering techniques can be evaluated. WK evaluates both the similarities and disagreements between pair-wise clustering arrangements in a matrix. This allows for agreements between different inputs to be produced. WK compares clusters that generate a score within the range -1.0 to $+1.0$ where -1.0 denotes no concordance and $+1.0$ denotes complete concordance between the clustering arrangements. The

distinction between the score establishes the structure of the arrangements. For example, a high WK value indicates that the two arrangements are very similar, whilst a low value indicates that they are dissimilar. A value close to 0.0 is usually observed for random clusters, indicating they are not similar and has no values in common. Table 2 shows the interpretation table for WK values.

Table 2. Agreement strength of Weighted-Kappa

Weighted-Kappa	Agreement strength
$0.0 \leq K \leq 0.2$	Poor
$0.2 < K \leq 0.4$	Fair
$0.4 < K \leq 0.6$	Moderate
$0.6 < K \leq 0.8$	Good
$0.8 < K \leq 1.0$	Very good

Some of the inputs and the datasets are poor for consensus clustering; however, it is difficult to easily identify the inputs and datasets that are poor as unsupervised learning has no gold standard to compare it to. Thus, this research work relies on the WK metric for the evaluation of the inputs. WK is implemented to measure the similarity between inputs for all of the selected datasets. An equivalent metric to the WK metric is Hubert-Arabie Adjusted Rand [16]. They can be both used in cluster analysis for comparing two clustering inputs obtained from different clustering methods. WK was selected for this study because of its similarity with Adjusted Rand and the benefit of having a qualitative interpretation, c.f. Table 2.

4 Defining the Threshold

As mentioned earlier, certain inputs and datasets can produce poor WK values. There is therefore a need to select an appropriate threshold value. Data that does not cluster are data that has an average WK value of less than that of a particular threshold. Even though the WK interpretation table, Table 2 displays a rough scale of what is expected for the agreement strength, another way was needed to define the threshold value. Thus, we conducted a simulation experiment which generated a million pairs of random clustering arrangements of different number of variables, n. The values of n start at 100 and increments by 100 each time until it reaches 1,000 (10 different sizes). Then, two random clusters are chosen and the WK values of these two clustering arrangements are recorded. This is repeated for all clustering arrangements produced. The simulations results were plotted and the distribution of WK is observed to find the most appropriate threshold. The maximum, minimum, average and standard deviation WK values of the million random clusters, for each of the 10 sets of n variables, are computed and displayed on a plot (Fig. 1). The plot clearly shows a decreasing trend of the maximum WK values as the number of variables increase - the maximum value for a million simulations is rapidly approaching zero. We assume that the same pattern would continue after $n =$ 1,000, continuing to get closer to zero. However, we believe it will never reach zero along the x-axis as the model seems to be asymptotic.

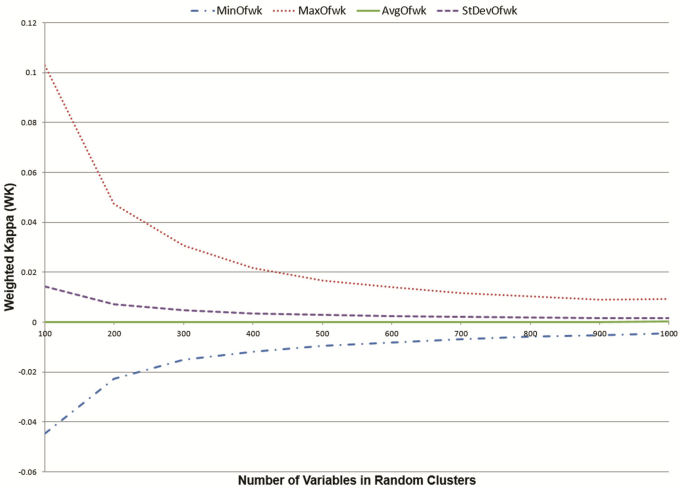

Fig. 1. Simulation experiment results plot of a million two random clustering arrangements of 10 varying number of variables, n.

From the simulations conducted it is clear that there is a very small chance of n variables to be above WK value of 0.1, as the maximum value starts off from approximately 0.1. In fact, the only time a WK value of 0.1 is attained was for the 100 variables run and that was only for one value out of a million simulations. Analysing the rest of the results from the simulations, it seems that this particular value is an outlier as the next maximum WK value achieved was for 0.08. There were only 12 points produced above the 0.08 threshold (out of a million simulations) and there are no points for the 0.09 WK value scale.

Since the WK value range is between $+1.0$ and -1.0, the minimum WK values generated from the simulation are negatives values. The minimum WK value seems to show the same trend as the maximum plot, except that it is on the opposite axis and that the negative values seem to be lower for the majority of the simulations. The average WK values produced from the simulations is around zero, as the simulations are producing small negative values as well as small positive values, this is indicated by the minimum and maximum plots. In addition, for the standard deviation plot, it seems that as the number of variables increases the range between them starts tightening down i.e., as the minimum and maximum plots converge, the standard deviation gets smaller. Both the mean and the standard deviation approach zero as the number of variables increase in size.

The average number of variables for the 198 datasets is 421. Thus, from the plot of the simulations, it can be seen that for 421 variables, the maximum WK value that can be produced is around 0.02. Anything above 0.02 is unlikely to occur at random. A WK threshold of 0.1 is still quite high, as the chance of two clustering arrangements that are being compared being random is extremely rare and if they are not random then they must be similar instead. However, the authors have decided to use it as the threshold

due to the maximum WK value produced from the simulations was 0.1 and also due to it being half-way between the poor scales of the agreement table (Table 2).

5 Matrix Creation and Problem Definition

For this work, 198 datasets were collated and 32 clustering techniques were selected as inputs. On an initial inspection of the datasets and inputs, it is clear that some of the datasets do not cluster well and some of the clustering methods are not as effective as others on the datasets. It is difficult to get representative datasets just by performing experiments on all datasets as they are all of different sizes and properties. The same difficulty can be said for the inputs (clustering methods).

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & \cdots & W_{1m} \\ W_{21} & W_{22} & \cdots & \cdots & W_{2m} \\ W_{31} & \vdots & & & W_{3m} \\ W_{41} & \vdots & & & W_{4m} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ W_{n1} & W_{n2} & \cdots & \cdots & W_{nm} \end{bmatrix}$$

Since all of the datasets under analysis contain the expected clustering arrangements, the techniques used in this work can be verified. To address this problem, a 198 × 32 matrix of the WK values of the inputs' (clustering methods) clustering arrangements versus the expected clustering arrangements for each of the datasets was constructed. Thus, let W be an n rows (number of datasets) by m columns (number of inputs) real matrix where the i^{th}, j^{th} value w_{ij} is the WK of input j (the actual clustering arrangement versus the expected clustering arrangement) applied to dataset i. Figure 2 displays a simplified representation of the matrix.

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & \cdots & W_{1m} \\ W_{21} & W_{22} & \cdots & \cdots & W_{2m} \\ W_{31} & \vdots & & & W_{3m} \\ W_{41} & \vdots & & & W_{4m} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ W_{n1} & W_{n2} & \cdots & \cdots & W_{nm} \end{bmatrix}$$

Fig. 2. Matrix representation of WK values (clustering inputs versus datasets).

To aid in the visualisation of the WK matrix (198 × 32), a heatmap of the WK values of the datasets and inputs was produced, shown in Fig. 3. An R package 'stats' (Version 3.5.0) was used to create the heatmap. WK values of 0.0 are shown in white (indicating

poor results) and WK values of 1.0 are shown in black (indicating identical clustering arrangements). Values between 0.0 and 1.0 are shown as shadows of grey. It can be clearly seen from the figure that many of the WK values of the inputs (compared to the actual clustering arrangements) for the datasets are poor. This indicates that some of the inputs do not cluster well on all of the datasets and that some of the datasets do cluster well at all. Thus, being able to identify the inputs and datasets that are poor and to exclude them from the matrix is important. The aim is to find the best balance between inputs and datasets. Manually removing the poor datasets or poor inputs would alter the row averages and column averages as they are interconnected. Thus, selecting the appropriate datasets and inputs becomes a sub-selection problem where the goal is to include as many datasets and as many clustering methods as possible.

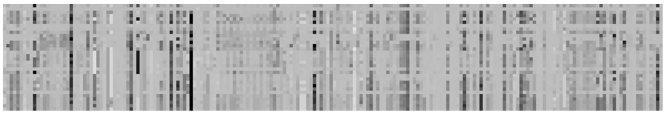

Fig. 3. A heatmap representation of the WK matrix.

6 Experimental Methods

As it is a subset selection problem and we look to maximise both datasets and clustering methods (inputs), we propose five methods for selecting the inputs and datasets. Two of the methods are based on manual selection (MS1 and MS2) and the other three are based on heuristic search techniques (Random Mutation Hill climbing, Simulated Annealing and Genetic Algorithms). The 198×32 matrix is used as input for the sub-selection algorithm and the output is a subset containing the selected datasets and inputs. As previously explained in Sect. 4, a WK threshold of 0.1 was selected and incorporated into the techniques proposed. The aim was to include as many datasets (rows) and clustering inputs (columns) as possible. The maximum number of datasets to include corresponds to the total number of datasets available and likewise for inputs.

6.1 Manual Selection Methods

The manual process involves removing all rows and all columns where the average WK value is less than 0.1. It is computed in two stages and thus two manual selection methods need to be defined. The WK threshold that was chosen for this work is 0.1. The first manual selection method (MS1) removes all rows that have an average WK value less than the threshold (step 1), then removes all columns that have an average WK value less than the threshold (step 2). The second manual selection method (MS2) removes all columns that are below the threshold (step 1) then removes all rows that are less than the threshold (step 2). Both methods will effectively reduce the original WK matrix in size. Results of both methods are displayed in Table 3.

Table 3. Sub-selection methods results table, displaying the number of included inputs and datasets for each of the proposed methods

Method	Datasets	Inputs	% Datasets	% Inputs
MS1	79	11	39.9	34.4
MS2	53	28	26.8	87.5
RMHC	60	28	30.3	87.5
SA	60	28	30.3	87.5
GA	60	28	30.3	87.5

6.2 Heuristic Search Methods

Three heuristic search methods were implemented - Genetic Algorithm, Hill Climbing and Simulated Annealing. For the three techniques selected a fitness function of quality was needed, defined in the next section. The same program with the same fitness function was used for the three algorithms. The fitness function simply works by creating a binary mask which corresponds to include or exclude from the matrix, such that 1 to include a dataset or an input from the matrix and vice versa for excluding a dataset or an input. Given a binary string, the number of rows and columns that is less than the threshold can be counted.

All three techniques were run for 10,000 iterations (or fitness function calls). The time it undertakes the program to run is proportional to the fitness function calls. For some of the experiments, the fitness function calls are referred to as the number of iterations and the runtime of the program is proportional to the iterations number. Either iterations or fitness function calls might be used throughout this paper. A well-known issue with heuristic search techniques is that they can get stuck at local maximums. One common solution to the problem is to restart the search at another random point. Thus, the three heuristic experiments were repeated 25 times each and the average was recorded.

$$\begin{aligned} f(S, W) &= \sum_{i=1}^{n} \sum_{j=1}^{m} \delta(s_i, 1)\delta(s_{j+n}, 1)(w_{ij} - T) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} \delta(s_i \times s_{j+n}, 1)(w_{ij} - T) \end{aligned} \tag{1}$$

Random Mutation Hill Climbing (RMHC) is a heuristic search algorithm that uses an iterative approach to find an objective in the search space by simply maximising the objective function. This algorithm starts at a random point in the search space and aims for a better fitness of the objective function by randomly searching the adjacent or closer neighbours. The process continues until the optimal solution is obtained and it starts all over again at the new point reached. The RMHC algorithm was chosen as it is the most basic variant of an evolutionary algorithm and it has been previously used in numerous studies such as [3].

Simulated Annealing (SA) is a meta-heuristic technique which improves on RMHC. The idea of SA originated from the natural process of annealing in metallurgy,

which involves heating materials to a very high temperature and then allowing it to slowly cool down to alter its physical structure. In SA a temperature parameter is kept to simulate the heating process (it expresses the probability of accepting a solution with a worst fitness). The temperature is initially set to a high value, allowing the temperature to steadily "cool" (decreasing whilst running the algorithm). This temperature keeps decreasing to reach a zero by the end of the algorithm, revealing the solutions. SA have been applied to various problems [20] and was chosen for this reason. We note that one per cent of the fitness function calls was used to find the starting temperature [28].

Genetic Algorithm (GA) was chosen as it is a powerful tool which can perform various optimisation problems. GA represents the solution to a problem as string (encoded as a chromosome). A population of chromosomes represents a subset of the search space of all possible solutions. The fitness function is also used to rate the worth of a solution that the chromosome represents. A standard binary GA using elitism was used. The population size of the GA was set to 25 and the generations were terminated after 10,000 fitness function calls. Genetic operations, mutation and crossover, are applied to the solutions to help find the best fitness. The mutation is set at 0.5/number of bits, whereas the crossover rate = 0.5.

6.3 Fitness Function

Let W be an n rows (number of datasets) by m columns (number of inputs) real matrix where the i^{th},j^{th} value w_{ij} is the WK of input j applied to dataset i. Given a binary string S of length $n + m$, the first n bits are a mask of what datasets are selected and the next m bits are a mask of what inputs have been selected. For example, if $n = 5$ and $m = 3$, then the string $S =$ "10101010" means that datasets 0, 2 and 4 have been selected along with input 1. Let s_i be the i^{th} bit of string S. The fitness function used in this paper is defined as follows:

The function $\delta(i, j)$ is the Kronecker delta function δ_{ij}, i.e. 1 if $i = j$, 0 otherwise.

Essentially the fitness function sums all of the WK values remaining after the binary string S has been applied, penalised by a threshold value T, 0.1. The rationale behind this fitness function is that when used in conjunction with a heuristic search method, a configuration of S will be found that maximises the number of WK values that are above T, excluding configurations where there are values below T. Alternative fitness functions were experimented with before selecting the current fitness function.

7 Results

Results from the experiments are shown in Table 3. It displays the number of inputs and datasets for each of the five proposed techniques. For all five methods, a threshold of 0.1 WK was used. Selecting a higher threshold would obtain worst results, fewer datasets and inputs. The table also displays the percentages of the inputs and datasets in proportion of the total number of inputs and datasets, respectively. From the table, it can be seen that MS1 produced the most datasets, 79, but much fewer inputs, 11. There is a large difference between the inputs for this method and the rest of the methods. As the

aim of the experiments is to include as many datasets and clustering inputs as possible, it seems that the three heuristic techniques produce the best results, 60 datasets and 28 inputs. However, as can be seen from the results all three heuristic methods produce the same optima, the same number of inputs and datasets. These identical results imply that the search space is relatively smooth and easy to search. In order to find the best technique, we need to analyse the three heuristic techniques in terms of runtime.

To find out the most efficient technique from the three heuristic methods, the convergence graphs of the three experiments were produced. As the experiments were repeated 25 times for each method, the average fitness is calculated and plotted on a graph (Fig. 4). From the plot, it can be seen that RMHC converges at the least number of iterations, 3,283, indicating that it is the most efficient in terms of runtime for this particular combinatorial optimisation problem. SA comes second in terms of runtime, converging at 5,701 iterations. Note that one per cent of the fitness function calls are sacrificed to find the initial temperature. GA performed the worst in terms of runtime converging the last out of the three methods. The authors presume if this was a larger feature subset selection problem, GA and/or SA might perform better.

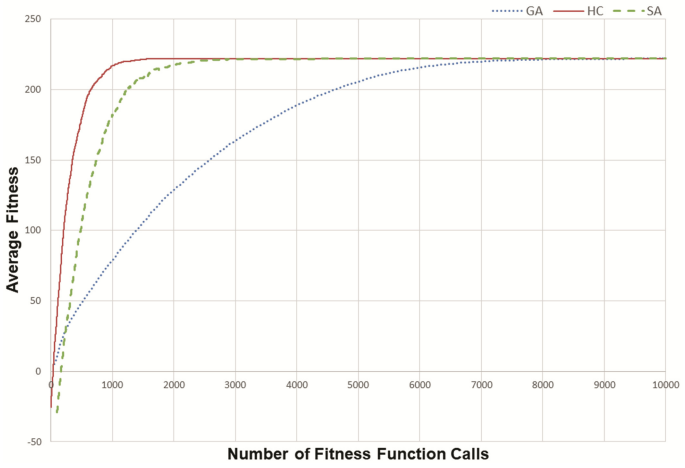

Fig. 4. A plot summarising the convergence points of GA, RMHC and SA.

The fitness function value produced from all three experiments is 221.895. The minimum theoretical fitness is −399.839 and the maximum theoretic fitness is 288.823. The fitness function value achieved is 90.2% of the way between the minimum and maximum theoretical fitness, points close to the minimum and maximum are almost certainly not achievable.

In addition, the three methods producing the exact same fitness function values does not mean that the outputted string binary arrangements are similar. Thus, we defined a simple metric based on Hamming Distance that cross-compares the similarity of each binary string representation from all the repeats to each other. It counts the occurrence of 1s and divides them by the number of possible 1s. This was computed for all three

heuristic search methods used in this study. Results show that the binary string representation produced from each of the 25 repeats of the three methods (75 in total) is identical to each other, illustrating consistency of the results.

8 Post-analysis Results

From the previous section RMHC was found to be the most efficient heuristic method for this particular combinatorial optimisation problem. As this technique outputs 60 datasets and 28 input clustering methods, a WK matrix of the 60 datasets versus the 28 inputs was constructed. The matrix is of the WK values of the output clustering arrangement versus the expected clustering arrangement of the 28 input methods, for the 60 datasets. The followings are post-analysis on the quality of the results.

A normality test was used to determine if the data under analysis was modelled by a normal distribution. It computes the likelihood of a random variable from the data to be of a normal distribution to find out how the averages of the inputs vary. Thus, to test for normality we averaged each column in the 60×28 matrix; this provided us with 28 values of 60 in size. Before testing for normality, we applied the following transformations to the data:

(a) Computed the absolute values of the results
(b) Removed any zero values
(c) Took the log (base e) of the results

Five normality tests were selected for this analysis, they are displayed in Table 4. They are part of the R "nortest" package (30 July 2015). The average results from these five normality tests are displayed in Table 4. By working out the average of the five normality tests, results show that 22 of the 28 inputs pass the normality test at the 0.01 significance level. Results show that the individual averages of the test values are normally distributed and so too the mean of the five tests. This indicates that it is possible to generate the input distribution from another normal distribution based on that mean i.e., by providing a distribution model to use based on WK values of the inputs.

Table 4. A summary of the normality test p-values and their mean for all inputs

Name	Test type	P-value
ad.test	Anderson-Darling Test for Normality	0.08062
cvm.test	Cramer-von Mises test of goodness-of-fit	0.07086
lillie.test	Lilliefors (Kolmogorov-Smirnov) Test for Normality	0.04948
pearson.test	Pearson Chi-Square Test for Normality	0.04352
sf.test	Shapiro-Francia Test for Normality	0.13560
Average		0.07601

For the 60 datasets, we computed the average WK of the inputs' clustering arrangements versus the expected clustering arrangements (we will refer to as inputs vs. expected). Subsequently, we computed the average WK of the inputs' clustering arrangements against each other's, again for the 60 datasets (we will refer to as inputs

vs. inputs). This provided us with two datasets of size 60. Correlating these two datasets produced 0.562 which shows a strong positive correlation above the 1% significance level. This suggests that if the number of inputs keeps increasing, the mean of the inputs vs. inputs will possibly start resembling the mean of the inputs vs. expected. From looking at the WK pairs of values (inputs vs. inputs and inputs vs. expected) it can be seen that in many of the cases they are close together. Thus, from the datasets selected, results show that the inputs vs. inputs are similar to the inputs vs. expected results, indicating a link between the intra-method agreement and the method versus the expected. This is interesting as it implies if we would like to find out the average quality of the inputs without having the expected clustering arrangements, we could evaluate the performance based on the average performance of the inputs vs. inputs. This allows us to have an approximate idea of the quality of the results (inputs) if there are no expected clustering arrangements to compare it with. We would only need to compare the input with every other input and compute the average. If it is very high we would know that the clustering methods are reasonably accurate and if it is very low we would know that the clustering methods are not appropriate. We look to further investigate this relationship as part of future work.

Moreover, we have computed the standard deviation of the inputs vs. inputs and inputs vs. expected. Correlating these together presents a correlation of 0.441, which clearly passes the 1% significance level. Lastly, to find out if there is a relationship between the average inputs vs. expected and the size of the datasets (number of instances and attributes), we correlated them. The correlation was shown to be 0.093 for the number attributes, and −0.165 for the number of instances. These two correlations are weak and they do not pass the 10% significance level (0.211). This indicates that there are no relationships between the average WK of the inputs and the changing dataset sizes. This relationship is independent i.e., does not increase or reduce based on the size of the dataset. The same was repeated for correlating the average inputs vs. inputs and the dataset sizes (number of instances and attributes). Correlations are also shown to be weak: 0.014 for the number of attributes and −0.187 for the number of instances. These results show that there is no bias in the WK values produced or our proposed data selection technique.

9 Conclusions and Future Work

Clustering inputs and datasets for consensus clustering can be poor. Moreover, it is not straightforward to identify the inputs that are poor or the appropriate datasets. This research work explores the behaviour of CC and looks at modelling a suitable distribution for it by selecting the most suitable inputs and datasets. Instead of selecting them at random, it proposed five sub-selection techniques (three heuristic and two manual selection methods) to achieve the largest number of inputs and datasets. Results showed that a normal distribution model, based on the WK values of the inputs, can be used to generate the input distributions of CC. Using these techniques may improve the results for CC. In addition, results have also presented a quick metric that can be used to estimate the quality of the inputs (clustering methods), if there are no expected clustering

arrangements to compare the outputted clustering arrangements with. For this work we are not applying the results to CC; however, we look to extend this study towards a larger issue of combinatorial optimisation and apply results to CC. For future work we would also seek to expand on the datasets; a larger number of dataset values is needed to model a more accurate distribution.

References

1. Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: ranking and clustering. J. ACM. **55**(5), 1–27 (2008). https://doi.org/10.1145/1411509.1411513
2. Altman, D.G.: Practical Statistics for Medical Research. Chapman and Hall, London (1997)
3. Arzoky, M., Swift, S., Tucker, A., Cain, J.: A seeded search for the modularisation of sequential software versions. J. Object Technol. **11**(2) (2012). https://doi.org/10.5381/jot.2012.11.2.a6
4. Azimi, J., Fern, X.: Adaptive cluster ensemble selection. In: IJCAI International Joint Conference on Artificial Intelligence, pp. 992–997 (2009)
5. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. Mach. Learn. **56**(1–3), 89–113 (2004). https://doi.org/10.1023/B:MACH.0000033116.57574.95
6. Caruana, R., Niculescu-Mizil, A., Crew, G., Ksikes, A.: Ensemble selection from libraries of models. In: Twenty-First International Conference on Machine Learning - ICML 2004, p. 18 (2004). https://doi.org/10.1145/1015330.1015432
7. Charikar, M., Guruswami, V., Wirth, A.: Clustering with qualitative information. J. Comput. Syst. Sci. **71**(3), 360–383 (2005). https://doi.org/10.1016/J.JCSS.2004.10.012
8. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Meas. **20**(1), 37–46 (1960). https://doi.org/10.1177/001316446002000104
9. Dua, D., Taniskidou, E.K.: UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, CA. http://archive.ics.uci.edu/ml. Accessed 6 Oct 2017
10. Dudoit, S., Fridlyand, J.: Bagging to improve the accuracy of a clustering procedure. Bioinformatics **19**(9), 1090–1099 (2003). https://doi.org/10.1093/bioinformatics/btg038
11. Fern, X.Z., Lin, W.: Cluster ensemble selection. Stat. Anal. Data Min. **1**(3), 128–141 (2008). https://doi.org/10.1002/sam.10008
12. Fern, X.Z., Brodley, C.E.: Solving cluster ensemble problems by bipartite graph partitioning. In: Twenty-First International Conference on Machine Learning - ICML 2004, p. 36 (2004). https://doi.org/10.1145/1015330.1015414
13. Fred, A.L.N., Jain, A.K.: Combining multiple clusterings using evidence accumulation. IEEE Trans. Pattern Anal. Mach. Intell. **27**(6), 835–850 (2006). https://doi.org/10.1109/TPAMI.2005.113
14. Gionis, A., Mannila, H., Tsaparas, P.: Clustering aggregation. In: Proceedings - International Conference on Data Engineering, pp. 341–352 (2005). https://doi.org/10.1109/ICDE.2005.34
15. Giotis, I., Guruswami, V.: Correlation clustering with a fixed number of clusters. In: SODA, p. 16 (2005). https://doi.org/10.1145/1109557.1109686
16. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**(1), 193–218 (1985). https://doi.org/10.1007/BF01908075
17. Hyndman, R.J.: Time series data library. http://data.is/TSDLdemo. Accessed 15 Oct 2017
18. Jain, K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. **31**(3), 264–323 (1999). https://doi.org/10.1145/331499.331504
19. Kaggle: Kaggle datasets. www.kaggle.com/datasets. Accessed 15 Sept 2017

20. Kirkpatrick, S., Gelatt, D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983). https://doi.org/10.1007/BF01009452
21. Kuncheva, L.I., Vetrov, D.P.: Evaluation of stability of k-means cluster ensembles with respect to random initialization. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11), 1798–1808 (2006). https://doi.org/10.1109/TPAMI.2006.226
22. Mldata.org.: Machine learning data set repository. http://mldata.org. Accessed 7 Dec 2017
23. Monti, S., Tamayo, P., Mesirov, J., Golub, T.: Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. Mach. Learn. **52**(1–2), 91–118 (2003). https://doi.org/10.1023/A:1023949509487
24. Pelleg, D., Moore, A.W.: X-means: extending k-means with efficient estimation of the number of clusters. In: Proceedings of the Seventeenth International Conference on Machine Learning. table contents, pp. 727–734 (2000). https://doi.org/10.1007/3-540-44491-2_3
25. Singh, V., Mukherjee, L., Peng, J., Xu, J.: Ensemble clustering using semidefinite programming with applications. Mach. Learn. **79**(1–2), 177–200 (2010). https://doi.org/10.1007/s10994-009-5158-y
26. StatLib.: StatLib—Datasets Azrchive. Carnegie Mellon University (1989). http://lib.stat.cmu.edu/datasets. Accessed 20 Nov 2017
27. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. **3**, 583–617 (2002). https://doi.org/10.1162/153244303321897735
28. Swift, S., Tucker, A., Vinciotti, V., Martin, N., Orengo, C., Liu, X., Kellam, P.: Consensus clustering and functional interpretation of gene-expression data. Genome Biol. **5**(11) (2004). https://doi.org/10.1186/gb-2004-5-11-r94

AQ4

Author Query Form

Book ID :	**473257_1_En**
Chapter No.:	**72**

Springer
the language of science

Please ensure you fill out your response to the queries raised below and return this form along with your corrections

Dear Author
During the process of typesetting your chapter, the following queries have arisen. Please check your typeset proof carefully against the queries listed below and mark the necessary changes either directly on the proof/online grid or in the 'Author's response' area provided below

Query Refs.	Details Required	Author's Response
AQ1	Kindly check and confirm the inserted e-mail address for the corresponding author.	
AQ2	Please check and confirm if the inserted publisher location is correct for the Ref. [2].	
AQ3	Please check and confirm the edit made in the author name for the Refs. [9, 24].	
AQ4	Reference [24] is given in the list but not cited in the text. Please cite this in text or delete this from the list.	

MARKED PROOF

Please correct and return this set

Please use the proof correction marks shown below for all alterations and corrections. If you wish to return your proof by fax you should ensure that all amendments are written clearly in dark ink and are made well within the page margins.

Instruction to printer	*Textual mark*	*Marginal mark*
Leave unchanged	··· under matter to remain	New matter followed by
Insert in text the matter indicated in the margin		or (x2)
Delete	through single character, rule or underline or through all characters to be deleted	or (x2)
Substitute character or substitute part of one or more word(s)	through letter or through characters	new character or new characters
Change to italics	under matter to be changed	
Change to capitals	under matter to be changed	
Change to small capitals	under matter to be changed	
Change to bold type	under matter to be changed	
Change to bold italic	under matter to be changed	
Change to lower case	Encircle matter to be changed	
Change italic to upright type	(As above)	
Change bold to non-bold type	(As above)	
Insert 'superior' character	through character or where required	or under character e.g. or
Insert 'inferior' character	(As above)	over character e.g.
Insert full stop	(As above)	
Insert comma	(As above)	
Insert single quotation marks	(As above)	or and/or or
Insert double quotation marks	(As above)	or and/or or
Insert hyphen	(As above)	
Start new paragraph		
No new paragraph		
Transpose		
Close up	linking characters	
Insert or substitute space between characters or words	through character or where required	
Reduce space between characters or words	between characters or words affected	